Real-time Domain Adaptation in Semantic Segmentation

Cristina Baitan, Cecilia Bergamini, Letizia Chiera, Rossana Ciarfaglia
Politecnico di Torino
s343428@studenti.polito.it, s343341@studenti.polito.it, s343342@studenti.polito.it,
s345926@studenti.polito.it

ABSTRACT

Semantic segmentation has recently achieved remarkable progress thanks to deep convolutional neural networks. However, training such models often requires large-scale annotated datasets, which are expensive and time-consuming to produce. To alleviate this issue, synthetic datasets have been proposed as a cost-effective alternative, though the performance of models trained on synthetic data typically degrades when evaluated on real-world images due to the domain gap. In this work, we investigate semantic segmentation using two representative architectures, DeepLabV2 and BiSeNet, trained on both synthetic (GTA5) and real (Cityscapes) datasets. We evaluate the models in terms of mean Intersection-over-Union (mIoU) and other performance metrics, analyzing the impact of training data source on segmentation accuracy. Furthermore, we explore the effectiveness of data augmentation strategies and domain adaptation techniques such as adversarial approach to mitigate the gap between synthetic and real domains. Our results show that while models trained solely on synthetic data underperform compared to those trained on real data, the integration of augmentation and adaptation methods significantly improves generalization, narrowing the performance gap and highlighting the potential of synthetic datasets for real-world semantic segmentation tasks.

1. INTRODUCTION

Semantic segmentation is a core task in computer vision, crucial for applications such as autonomous driving, robotic navigation, and detailed urban scene analysis. The objective is to assign a label to each pixel in an image, allowing models to capture fine-grained structural and semantic informations.

Despite the impressive results achieved by deep convolutional neural networks, their generalization often deteriorates when applied to data that differ from the training distribution. This performance drop, commonly referred to as a domain shift, presents a significant challenge when models trained on synthetic data are deployed in real-world environments.

To address this limitation, Unsupervised Domain Adaptation techniques aim to bridge the gap between a labeled source domain, such as synthetic datasets from GTA5, and an unlabeled target domain, like real images from Cityscapes. The issue is particularly pressing for semantic segmentation, as obtaining pixel-wise annotations for large-scale real datasets is prohibitively expensive and labor intensive.

This paper focuses on real-time semantic segmentation. At the beginning, we train DeepLabV2 and BiSeNet on the Cityscapes dataset. Then, we apply domain shift training on GTA5 and validating on Cityscapes. To overcome discrepancy, we introduce data augmentation strategy and then deal with an adversarial approach. Then, we experiment different combinations of optimizer and loss function.

2. DEEPLABV2 AND BISENET MODELS

In this paper we consider two broadly adopted architectures: DeepLabV2 and BiSeNet. In particular, The first one is an atrous convolution-based semantic segmentation network, while the second is a real-time bilateral semantic segmentation network.

2.1 DeepLabV2 model

The DeepLabV2 [2] architecture has become a benchmark in semantic segmentation due to its integration of atrous (dilated) convolutions together with the Atrous Spatial Pyramid Pooling (ASPP) mechanism. By aggregating features at multiple receptive fields, ASPP provides rich contextual cues while preserving spatial resolution, which is vital for dense prediction tasks. In our study, we employ a ResNet-101 backbone-initialized with ImageNet pre-training. At first, we train DeepLabV2 and validated on the Cityscapes dataset. This setup yields competitive results in terms of mean Intersection over Union (mIoU), though it demands considerable computational resources.

2.2 BiSeNet model

BiSeNet [6] is tailored for real-time semantic segmentation, aiming to balance efficiency and accuracy. The architecture is built around two key modules: the Spatial Path, which maintains detailed spatial resolution, and the Context Path, which extracts semantic information through a lightweight backbone combined with global pooling. For our work, we adopt BiSeNet with a ResNet-18 backbone pre-trained on

ImageNet. This network is employed in all experiments related to domain shift and adaptation, as it offers fast inference and requires, comparatively to DeepLabV2, fewer computational resources.

3. TRAINING PHASE

3.1 Loss function

As semantic segmentation is a multi-class classification problem for each pixel, during training phase we use a cross-entropy-loss, applying it to each pixel rather than to the whole image. Naming C the number of labels and N the total of pixels:

$$L_{main} = -\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c} log(\tilde{y}_{i,c}^{main})$$

$$y_{i,c} = \begin{cases} 1 \text{ if } c \text{ correct label for } x_i \\ 0 \text{ otherwise} \end{cases}$$

$$\tilde{y}_{i,c}^{main} = \frac{exp(x_{i,c})}{\sum_{k \in C} exp(x_{i,k})}$$

We also apply auxiliary losses to intermediate layers of a neural network. They provide extra gradient signals during training, helping to stabilize learning, accelerate convergence, and encourage the model to learn more robust and general feature representations. Supposing to have M auxiliaries outputs whose prediction is $\tilde{y}_{i,c}^{aux_m}$, L_{aux_m} is the respective auxiliary loss and:

$$L_{total} = L_{main} + \beta \sum_{m=1}^{M} L_{aux_m}$$

We set $\beta = 0.4$ taking inspiration from [7].

3.2 Optimization setup

We provide training exploiting Stochastic Gradient Descent (SGD) including momentum and weight decay. Specifically, we set the momentum coefficient $\rho = 0.9$ and the weight decay $\lambda = 5 \times 10^{-4}$ as used in [2].

$$v_{t+1} = \rho v_t - \alpha_t(\nabla f(x_t) + \lambda x_t)$$
$$x_{t+1} = x_t + v_{t+1}$$

During training, we apply the polynomial *learning rate* policy because we have observed that it was more effective than the step learning rate.

$$\alpha(t) = \alpha_0 \left(1 - \frac{t}{max\_iter}\right)^p$$

where $\alpha_0 = 2.5 \times 10^{-2}$, $max\_iter = 300$ and power $p = 0.9$ as proposed in [6].

3.3 Automatic Mixed Precision

We add the Automatic Mixed Precision (AMP) technique to speed up the training process and diminish memory usage. GradScaler, utility specific to AMP, is used to prevent numerical underflow when computing gradients in mixed precision, ensuring stable and accurate model

We consider all pixels when training the network, labeling as *void class* those different from the 19 labels considered. The choice was driven by the intention to allow the model to capture the full context of the scene, including elements that are unfamiliar or not labeled, with the goal of enhancing its effectiveness in real-world scenarios.

3.4 Datasets and preprocessing

The illustrated results were obtained using the following datasets.

- The ***Cityscapes*** [3] dataset is a high-resolution (1024 x 2048) collection of urban street scenes for semantic segmentation. It includes 20,000 coarsely and 5,000 finely annotated images, with the fine set divided into training, validation, and test. The total number of classes is 30. Of these, 19 classes are considered, while the remaining classes are treated as *void* and ignored in the evaluation of the model. The images were captured in varying months and daytime conditions and selected to cover diverse objects, scene layouts, and backgrounds.
- The ***GTA5*** [4] dataset comprises 24,966 images at a resolution of 1914 × 1052, synthetically generated. Its ground-truth annotations align with the 19 categories defined in the Cityscapes dataset.

Before training some preprocessing operations are done:

- Resizing: both images and labels are resized to smaller dimensions, half of the original size for the Cityscapes dataset for example, to reduce memory usage.
- Normalizing: each pixel is replaced by its normalized value, using the mean and standard deviation computed over all pixels of the dataset for each channel. In this case, the ImageNet mean and standard deviation are used, since the backbone was pretrained on ImageNet.

4. EXPERIMENTAL EVALUATION

4.1 Evaluation Metrics

We assess the performance in terms of mean Intersection over Union (mIoU) on the 19 standard semantic categories, along with measurements of latency, FLOPs, and parameter count.

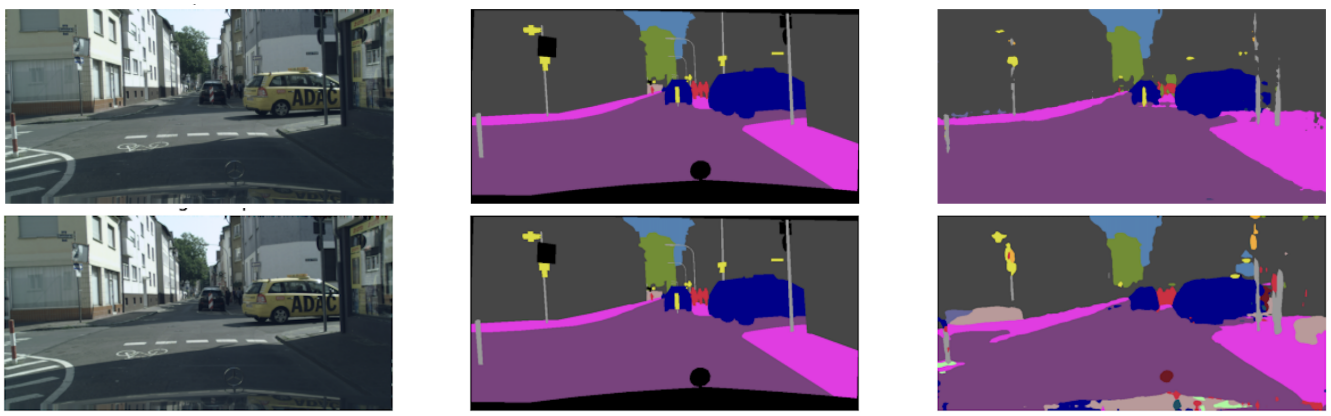
$$mIoU = \frac{1}{C}\sum_{c=1}^{C} IoU_c \qquad IoU_c = \frac{TP_c}{TP_c + FP_c + FN_c}$$

We also record the training loss at each epoch to check the progress and ensure that the training is proceeding correctly.

4.2 Testing Semantic Segmentation Networks

The two models, DeepLabV2 and BiSeNet, were trained on the Cityscapes dataset for 50 epochs, using an input resolution of 512×1024 pixels for both training and testing. The first one has R101 as backbone while the second uses ResNet18, both pre-trained on ImageNet.

We train Bisenet imposing batch size equal to 4 and we set the parameter $num\_workers = 2$ in order to use 2 parallel CPU processes to load data batches. On the other hand,



(a) Input (b) Ground Truth (c) Prediction

Figure 1: Results produced by DeepLabV2 (first row) and BiSeNet on Cityscapes (second row).

Model	MIoU(%)	Latency(ms)	FLOPs(G)	Params(M)
DeepLabV2	51.77	247.23 ± 12.31	375	43.901
BiSeNet	45.39	16.20 ± 1.35	25.78	12.582

Table 1: Metrics for DeepLabV2 and BiSeNet. Training and validation on Cityscapes dataset

DeepLab V2 is trained imposing a very little batch size, equal to 2, due to the limitations of memory and GPU.

Results are illustrated in Table 1 and Figure 1. As expected, we achieve better performance with DeepLabV2, with a final mIoU higher of 6 point percentage with respect to BiSeNet, even though the second reports a significant reduction in terms of latency, FLOPs and parameters. This is exploited by the fact that BiSeNet performs better with real-time tasks. In contrast, DeepLab V2 is a more complex model which require more effort in term of computational but delivers better performance.

4.3 Domain Shift

In this section we use BiSeNet as our segmentation architecture and consider as upper bound the results obtained in Table 1 regarding Real-time Cityscapes.
We train the model on GTA5 with Backbone ResNet18 and then Cityscapes as validation set .

Comparing values, we observe a decrease in performance when testing the network on the target domain, the mIoU drops from 45.39% to 18.76%. This underlines how semantic segmentation models are strongly affected by differences in surface patterns, illumination, chromatic balance, and the general look and feel when comparing synthetic images with real ones.

Results are reported in Table 2 in terms of mIoU for the 19 classes considered and Fig. 2(c) shows the prediction with respect to the ground truth. As expected, higher performance is observed for classes that remain visually consistent across domains, such as road, vegetation, and sky, since their features (e.g., shape, texture, and color) are relatively invariant and undergo little change when moving across datasets. On the contrary, the model performs worse with less frequent categories (e.g. train, bicycle). These classes are context and instance dependent, aspects that the model has difficulties to understand without adaptation.

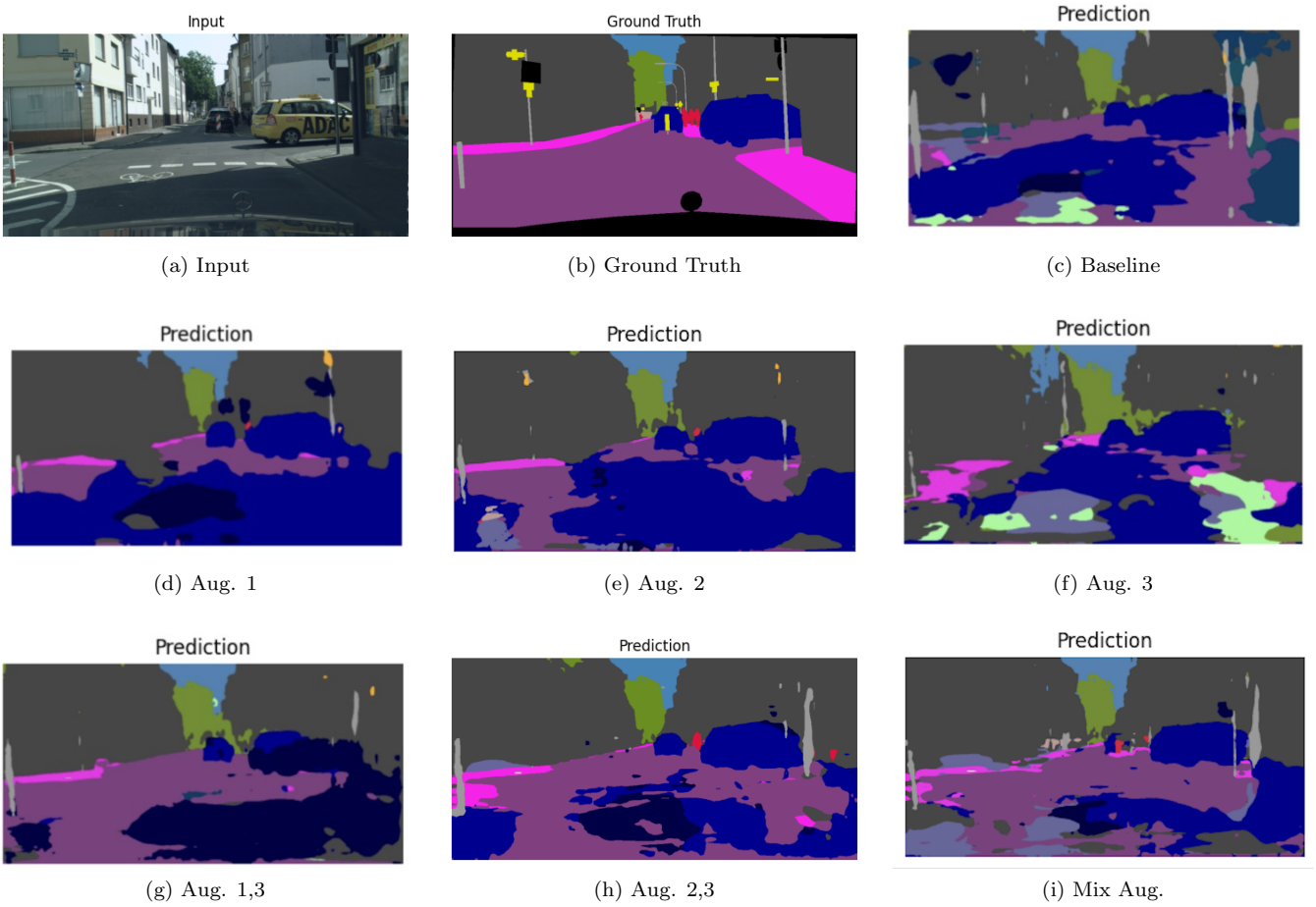
4.4 Data augmentation to reduce domain shift

We introduce data augmentations during training to improve the generalization capability of the segmentation network trained on GTA5.
We apply three different augmentations, and their combinations, each with a probability $p = 0.5$, to enrich the discrepancy between training and testing data, to build a model able to deal with variations between synthetic and real-world images. In particular:

- Aug. 1: **Horizontal Flip Flop**: introduces left–right symmetry by mirroring the image, encouraging the model to learn orientation-invariant features.
- Aug. 2: **Color Jitter**: perturbs brightness, contrast, saturation, and hue, simulating variations in illumination and color distribution across domains.

	MIoU (%)	road	sidewalk	building	wall	fence	pole	traffic light	traffic sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motorcycle	bicycle
Baseline	18.76	44.8	7.02	54.42	2.40	3.81	20.41	11.32	4.46	65.21	6.56	69.01	21.29	4.76	37.42	2.30	0.45	0.30	0.42	0.0

Table 2: Domain Shift GTA5 → Cityscapes: per-class IoU and mean IoU (%) on Cityscapes dataset.



(a) Input (b) Ground Truth (c) Baseline

(d) Aug. 1 (e) Aug. 2 (f) Aug. 3

(g) Aug. 1,3 (h) Aug. 2,3 (i) Mix Aug.

Figure 2: Comparison of predictions from different augmentations.

	MIoU (%)	road	sidewalk	building	wall	fence	pole	traffic light	traffic sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motorcycle	bicycle
Aug. 1	16.25	29.97	11.99	51.68	5.42	1.26	8.52	7.65	1.66	54.47	13.49	65.45	27.40	0.49	23.37	1.30	4.29	0.0	0.38	0.0
Aug. 2	23.07	46.57	26.49	71.91	7.59	7.17	13.23	9.10	2.03	76.26	23.10	82.42	31.75	4.28	26.32	5.28	0.91	0.02	3.85	0.0
Aug. 3	15.71	18.51	15.60	45.22	3.21	1.68	8.14	3.18	3.31	62.14	5.83	61.49	24.55	0.69	29.63	6.20	5.04	0.52	3.50	0.0
Aug. 1,2	22.79	52.79	10.07	66.90	10.46	9.05	17.95	15.61	4.09	71.34	21.87	78.36	25.42	1.61	43.45	1.05	0.11	0.0	2.90	0.0
Aug. 2,3	25.57	78.61	17.23	70.40	17.30	7.76	19.27	7.39	1.35	72.95	14.09	75.19	33.04	7.38	51.49	4.90	1.28	0.10	6.14	0.0
Mix 1,2,3	24.65	83.62	14.70	67.33	10.47	6.73	16.26	6.15	2.28	73.08	28.03	67.86	24.91	2.88	53.70	5.83	0.02	0.0	4.56	0.0

Table 3: Augmentation GTA5 → Cityscapes: per-class IoU and mean IoU (%) on Cityscapes dataset.

- Aug. 3: **Gaussian Blur**: applies a smoothing kernel to the image, replicating effects such as defocus or motion blur and enhancing robustness to image quality degradation.

The results are illustrated in Table 3 and Figure 2. The best-performing combinations are respectively Color Jitter and Gaussian Blur, *Aug. 2,3*, and the set of the three augmentations proposed, *Mix Aug.* Comparing with baseline results obtained by training the model on the classical GTA5 in Table 2, we observe that all augmented models, except for the single Horizontal Flip Flop, *Aug. 1*, have reached an improvement in mIoU during the validation phase, this suggests that applying a variety of visual perturbations improves the model's ability to generalize to real-world data. The most significant progresses were achieved when including Color Jitter, either alone or combined with other augmentations. The Color Jitter transformation enhances the model's robustness to color variations, improving its generalization across different visual conditions. This proved particularly effective in our setting, where the model was trained on the GTA5 dataset and tested on Cityscapes, two datasets exhibiting meaningful domain shifts in color distribution and illumination.

	MIoU (%)	road	sidewalk	building	wall	fence	pole	traffic light	traffic sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motorcycle	bicycle
Adversarial	23.89	80.88	22.87	63.72	15.24	8.40	11.26	3.79	4.98	70.47	14.42	43.71	22.71	0.0	68.10	12.16	6.11	1.97	3.16	0.0

Table 4: Adversarial GTA5 → Cityscapes: per-class IoU and mean IoU (%) on Cityscapes dataset.

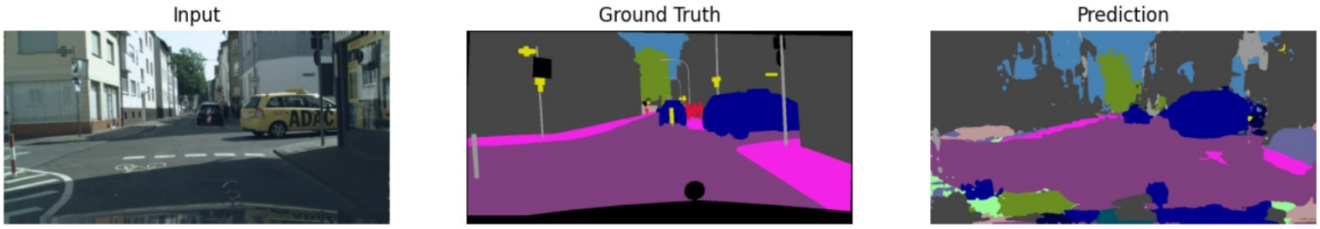


Figure 3: Results obtained applying adversarial method.

4.5 Adversarial approach in domain adaptation

To tackle the problem of domain shift, we introduce an adversarial approach. The main idea is to create a competition between two networks: the segmentation network, which we aim to train, and another network called the discriminator. The discriminator's task is to recognize whether the output of the segmentation network comes from an input in the source domain or the target domain. Meanwhile, the segmentation network is trained not only to classify pixels correctly but also to "fool" the discriminator. The discriminator is implemented using a single-level adversarial learning. Training of the segmentation network and the discriminator is performed jointly in a single stage, using the Adam optimizer for the discriminator and SGD for the segmentation network following the architecture proposed in [5]. Adam's parameters and the initial learning rate are fixed as proposed in [5].
The loss function used to train the discriminator is Binary Cross Entropy:

$$\sigma(z_i) = \frac{1}{1+e^{-z_i}}$$

$$\ell(z_i, y_i) = -\big[y_i \cdot \log(\sigma(z_i)) + (1-y_i) \cdot \log(1-\sigma(z_i))\big]$$

$$\mathcal{L}(\mathbf{z}, \mathbf{y}) = \frac{1}{N}\sum_{i=1}^{N} \ell(z_i, y_i)$$

- y_i is the actual binary label (0 or 1) of the i^{th} observation.
- z_i is the predicted output of the i^{th} observation in class 1.

The loss function used to train the segmentation network is a weighted sum of the Cross Entropy loss (illustrated in Section 3) and an adversarial loss designed to fool the discriminator:

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

where $\lambda_{\text{adv}} = 0.001$ as in [5].
The reported results assume the use of the synthetic labeled source dataset GTA5 and the real-world unlabeled target dataset Cityscapes. For this experiment, we applied the best configuration obtained from the data augmentation evaluation (Gaussian Blur + Color Jitter) to reduce domain shift. The results, shown in Table 4 and Figure 3, report a mean IoU of 23%. This is a good result compared to experiments on domain shift without data augmentation, and it is similar to the best configuration achieved using data augmentation. Compared to the latter, we also observe variations in per-class IoU: some classes are better segmented using the adversarial approach, such as traffic signs, cars, and trucks, while others perform worse, such as poles, sky, riders, and persons.

4.6 Extension: Focal Loss and Adam optimizer

In this section, we propose additional techniques to further enhance performance in the domain adaptation task, including the evaluation of alternative loss functions such as Focal Loss and different optimizers, such as Adam.

$$L_{Focal} = -\sum_{i=1}^{N}\sum_{c=1}^{C}(1-\tilde{y}_{i,c})^{\gamma} y_{i,c} log(\tilde{y}_{i,c})$$

Where γ is a term that gives more importance to missclassified examples. Specifically, the values experimented are $\gamma \in \{0, 2, 3\}$ to analyze the behavior of focal loss under different levels of emphasis on hard examples. The case $\gamma = 0$ corresponds to the standard cross-entropy loss and serves as a baseline. Values $\gamma = 2$ and $\gamma = 3$ are commonly used in the literature on focal loss and progressively increase the focus on difficult or misclassified pixels. This choice allowed us to study how stronger re-weighting affects convergence and segmentation performance in the domain adaptation setting.

4.6.1 Effects of Focal Loss using SGD

First, we apply Focal loss [1] with optimizer SGD.

γ	last mIoU(%)	best mIoU(%)	epoch best mIoU
0	25.57	25.90	31
2	22.53	25.07	36
3	21.03	22.21	49

Table 5: Final mIoU on Cityscapes tuning γ parameter using as optimizer SGD.

When using SGD as the optimizer, we observed that the standard cross-entropy loss ($\gamma = 0$) achieved the best results, with a maximum mIoU of 25.9%. Looking at Table 5, increasing γ to 2 or 3, which corresponds to emphasizing harder pixels through focal loss, led to lower overall performance and slower convergence. In particular, the best epochs shifted to later stages of training (36 and 49 for $\gamma = 2$ and $\gamma = 3$, respectively), indicating that the optimizer required more iterations to reach its peak. However, even with longer training, the final mIoU values for higher γ remained consistently below those achieved with the baseline cross-entropy loss. These results suggest that in our domain adaptation setting, focal loss with higher γ may overemphasize noisy or uncertain pixels thereby reducing the benefit of SGD's more stable gradient updates.

Figure 4. illustrates qualitative segmentation results when training with different values of the focal loss parameter γ. The second row shows the ground truth annotations, where large regions such as road (dark purple), sidewalk (magenta pink), building (grayish violet), and vegetation (olive green) are clearly distinguishable, alongside smaller classes like traffic sign (yellow) and person (dark red), see Table 6 to understand color mapping.
With $\gamma = 0$ (third row, equivalent to standard cross-entropy), the predictions capture the main structural elements of the scene, correctly segmenting the road and vegetation areas and providing reasonable outlines for buildings. However, smaller objects such as traffic lights (orange) and traffic signs (yellow) are often missed or confused, while thin structures like poles (light gray) are rarely detected.
When increasing to $\gamma = 2$ (fourth row), the model puts stronger emphasis on difficult pixels. This results in noisier predictions: large homogeneous areas such as the road and sky become fragmented. Some small objects occasionally appear, but at the cost of deteriorated consistency in dominant classes.
With $\gamma = 3$ (fifth row), the effect is amplified. The segmentation maps are heavily degraded, with most pixels misclassified into a few dominant classes such as road and vegetation, while other classes nearly disappear. In particular, buildings (grayish violet) and sidewalks (magenta pink) are poorly recovered, and rare categories like person or bicycle vanish completely.
Overall, these results confirm the quantitative findings: while focal loss with higher γ emphasizes hard pixels, in this domain adaptation setting it tends to amplify noise and label uncertainty, leading to less stable and less accurate predictions compared to the baseline cross-entropy loss.

trainId	Color (RGB)	Descriptive color	Class
0	(128, 64, 128)	dark purple	road
1	(244, 35, 232)	magenta pink	sidewalk
2	(70, 70, 70)	dark gray	building
3	(102, 102, 156)	grayish violet	wall
4	(190, 153, 153)	beige pink	fence
5	(153, 153, 153)	light gray	pole
6	(250, 170, 30)	orange	traffic light
7	(220, 220, 0)	yellow	traffic sign
8	(107, 142, 35)	olive green	vegetation
9	(152, 251, 152)	light green	terrain
10	(70, 130, 180)	sky blue	sky
11	(220, 20, 60)	dark red	person
12	(255, 0, 0)	bright red	rider
13	(0, 0, 142)	dark blue	car
14	(0, 0, 70)	very dark blue	truck
15	(0, 60, 100)	teal blue	bus
16	(0, 80, 100)	dark teal	train
17	(0, 0, 230)	bright blue	motorcycle
18	(119, 11, 32)	burgundy	bicycle

Table 6: Color mapping of Cityscapes classes.

4.6.2 *Effects of Focal Loss using Adam optimizer*

In addition, we tested the Adam optimizer as an alternative to SGD. While SGD with momentum is widely used for segmentation, Adam provides adaptive learning rates that can stabilize training when the loss landscape is more complex. Moreover, considering the domain shift between GTA5 (synthetic) and Cityscapes (real), Adam can help accelerate convergence and potentially mitigate optimization difficulties.

$$
\begin{aligned}
m_{t+1} &= \beta_1 m_t + (1-\beta_1)\nabla f(x_t) \\
v_{t+1} &= \beta_2 v_t + (1-\beta_2)(\nabla f(x_t))^2 \\
\hat{m}_{t+1} &= \frac{m_{t+1}}{1-\beta_1^{t+1}} \\
\hat{v}_{t+1} &= \frac{v_{t+1}}{1-\beta_2^{t+1}} \\
x_{t+1} &= x_t - \alpha_t \frac{\hat{m}_{t+1}}{\sqrt{\hat{v}_{t+1}} + \epsilon}
\end{aligned}
$$

where m_t plays the role of the adaptive momentum, it estimates the running average of the gradient, v_t accumulates the second-order information and the corrections $\hat{m}_t$, $\hat{v}_t$ are used to remove the initialization bias.
In particular, we set $(\beta_1, \beta_2) = (0.9, 0.99)$.

γ	last mIoU(%)	best mIoU(%)	epoch best mIoU
0	9.39	16.95	26
2	11.31	20.46	9
3	16.94	24.67	11

Table 7: Final mIoU on Cityscapes tuning γ parameter using as optimizer Adam.

The results obtained with the Focal Loss using the Adam optimizer (Table 7) show that while Adam enables rapid convergence, reaching the best mIoU within the first epochs, the final performance tends to drop significantly by the end of training. Higher values of the focusing parameter γ improve the peak mIoU (up to 24.67% for $\gamma = 3$), indicating that the network benefits from emphasizing harder examples during training. However, the last epoch mIoU remains substantially lower than the best, suggesting instability or overfitting when using Adam in combination with Focal Loss.

In contrast, when using SGD with Focal Loss (Table 5), the network converges more slowly but achieves higher and

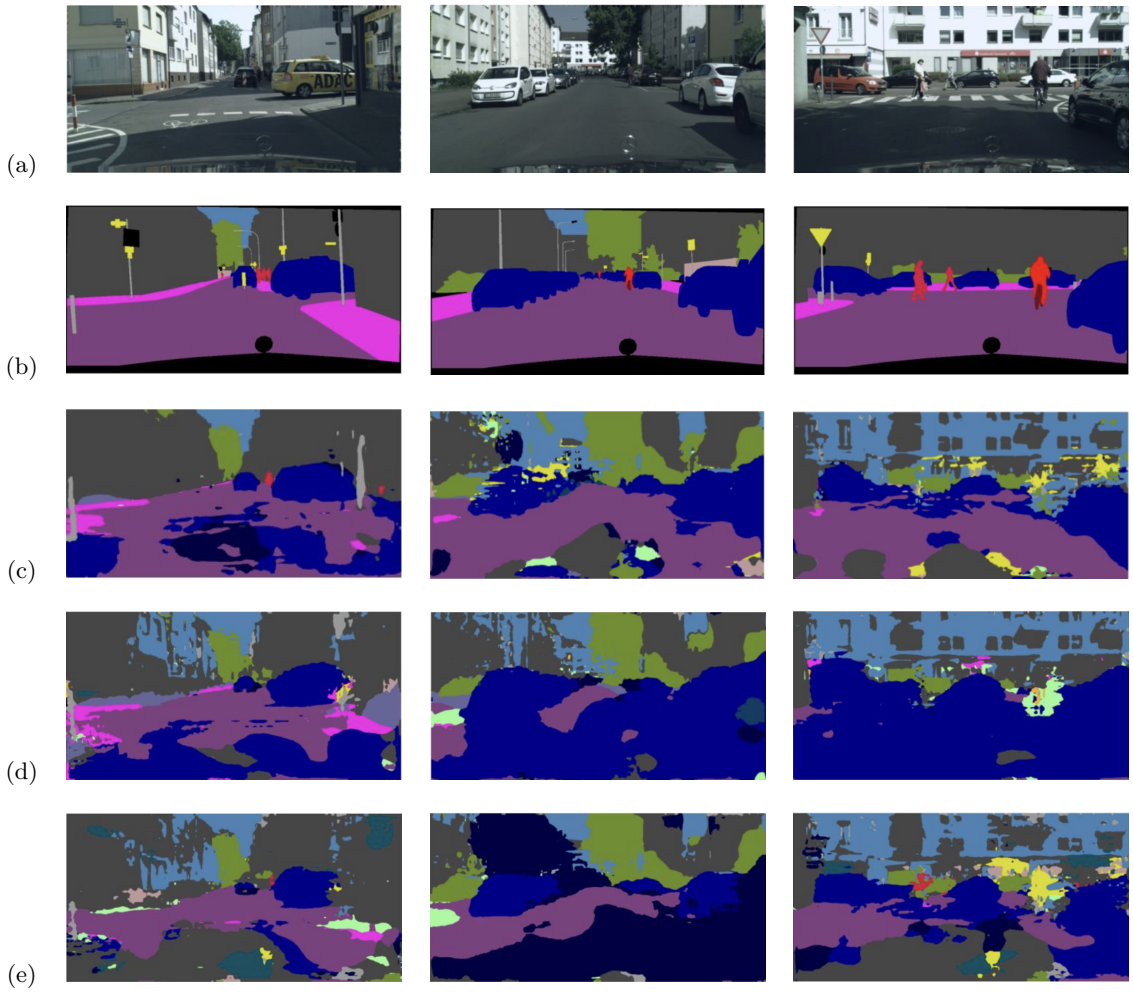


Figure 4: Qualitative segmentation results on GTA5 to Cityscapes when adopting focal loss with different γ. We present (a) Target Image, (b) Ground Truth, (c) focal loss with $\gamma = 0$, (d) focal loss with $\gamma = 2$, (e) focal loss with $\gamma = 3$

Table 8: Per-class IoU and mIoU, Focal Loss ($\gamma = 2$) and SGD on Cityscapes validation set

	mIoU(%)	road	sidewalk	building	wall	fence	pole	traffic light	traffic sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motorcycle	bicycle
FC γ=2 + SGD	26.31	64.11	24.44	62.85	11.77	11.60	12.19	2.72	2.58	72.79	19.95	44.88	32.23	2.83	31.55	7.98	21.26	1.21	1.07	0.04

more stable performance, with the final mIoU remaining close to the peak values across all γ settings. This indicates that SGD better preserves generalization, especially in the presence of domain shift between the synthetic training set (GTA5) and the real test set (Cityscapes). Overall, these results highlight a clear trade-off: Adam accelerates early convergence but may suffer from instability with Focal Loss, whereas SGD provides more robust and consistent performance over the full training schedule. This behaviour can be seen in Figure 5 that shows loss function evaluate on validation dataset Cityscapes through epochs.

Moreover, Adam tends to favor higher γ values, achieving its best results at $\gamma = 3$, whereas SGD obtains its optimal performance with lower γ, specifically at $\gamma = 0$.

4.6.3 *Comparison of Accuracy: Cross-Entropy vs Focal Loss*

A comparison of per-class IoU (for instance, by examining Table 4 against Table 8) reveals the impact of using Focal Loss with SGD ($\gamma = 2$) versus Cross-Entropy with SGD. In particular, while the accuracy for dominant classes such as road slightly decreases with Focal Loss, the segmentation of rare classes, including bus and rider, improves significantly. On the other hand, some moderately frequent classes, like traffic sign and car, achieve higher IoU with Cross-Entropy. This behavior can be interpreted in terms of how Focal Loss

works: by design, Focal Loss reduces the weight of well-classified examples and focuses more on hard-to-classify examples.

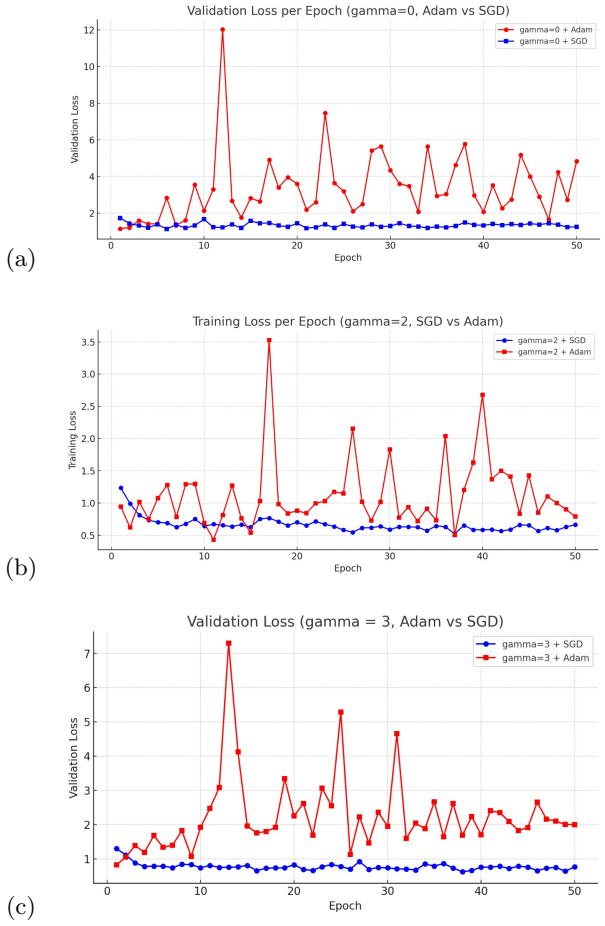


Figure 5: Comparison of loss curves when testing focal loss with SGD and Adam varying γ parameter. We illustrate (a) $\gamma = 0$, (b) $\gamma = 2$, (c) $\gamma = 3$.

4.6.4 Training with Focal Loss: Adam vs SGD

When using Adam, validation losses exhibit strong oscillations across epochs, see Figure 5. This behavior suggests that while Adam adapts quickly to the gradient signals, it struggles to maintain stability under the reweighting effect of the Focal Loss, leading to inconsistent optimization dynamics. In contrast, SGD produces smoother and more gradually decreasing loss trajectories, with reduced variance between training and validation curves. This indicates that SGD interacts more robustly with the Focal Loss, mitigating overfitting and better preserving generalization to the target domain. Overall, the loss dynamics confirm the trade-off observed in the mIoU results: Adam achieves faster but unstable convergence, whereas SGD ensures steadier and more reliable training progress.

5. CONCLUSIONS

In this work, we analyzed the problem of semantic segmentation in real-time scenarios, focusing in particular on the challenge of domain shift between synthetic and real data. The initial training on the Cityscapes dataset confirmed the potential of the DeepLabV2 and BiSeNet architectures, with approximately 52% and 43% mIoU respectively, highlighting how the former achieves superior performance in terms of mIoU, while the latter proves to be lighter and more efficient for real-time applications.

The experiments trained on GTA5 and tested on Cityscapes revealed a performance drop due to the domain gap between the datasets, with a clear decrease in mIoU down to 18.76%. To address this issue, data augmentation strategies and adversarial adaptation methods were employed, resulting in a significant improvement in the model's generalization ability. In particular, the combination of Color Jitter and Gaussian Blur transformations, related respectively to photometric and spatial variation, proved to be the most effective, achieving a mIoU of about 25.5%. Additionally, the adversarial adaptation approach yielded promising results, reaching a mIoU of 23%.

Finally, the use of Focal Loss combined with two different optimizers showed that SGD ensures greater stability and more consistent performance compared to Adam, which in this case resulted in less robustness.

Overall, the results obtained indicate that the combination of targeted data augmentation and domain adaptation techniques represents an effective direction to make this model applicable in concrete applications.

6. REFERENCES

[1] R. Azad, M. Heidari, K. Yilmaz, M. Hüttemann, S. Karimijafarbigloo, Y. Wu, A. Schmeink, and D. Merhof. Loss functions in the era of semantic segmentation: A survey and outlook. *arXiv preprint arXiv:2312.05391v1*, 2023.

[2] L.-C. Chen, G. Papandreou, K. Murphy, and A. 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.

[3] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[4] S. R. Richter, V. Vineet, S. Roth, and V. Koltun. Playing for data: Ground truth from computer games. In B. Leibe, J. Matas, N. Sebe, and M. Welling, editors, *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, 2016.

[5] Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, and M. Chandraker. Learning to adapt structured output space for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7472–7481, 2018.

[6] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *European Conference on Computer Vision (ECCV)*, 2018.

[7] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2881–2890, 2017.

APPENDIX

Link to the code folder: https://drive.google.com/drive/folders/1r_2-PZmzru-d9C5doJulKcB0Kd0OldGq?usp=share_link