Real-time Domain Adaptation in Semantic Segmentation

Milena Yahya
Politecnico di Torino
Turin, Italy
s306978@studenti.polito.it

Alexandro Buffa
Politecnico di Torino
Turin, Italy
s316999@studenti.polito.it

Simone Giambrone
Politecnico di Torino
Turin, Italy
s317002@studenti.polito.it

***Abstract*—Semantic Segmentation has numerous real-world applications including but not limited to medical imaging, autonomous driving, robot vision, and object detection. Thus, the demand on this technology is continuously rising. However, it requires a large amount of pixel-wise annotated images during the training phase, and pixel-wise annotating real-life datasets requires a lot of time and human effort. Due to the high labor cost of annotating segmentation ground truth, there has been great interest in employing large-scale synthetic datasets with annotations instead. In this paper, we implement a semantic segmentation network, then we explore different Domain Adaptation techniques to close the domain gap between the synthetic dataset used for training, and the target dataset which is a real-world dataset. We report our results and compare them using performance metrics like mIoU and pixel precision. Our experiments demonstrate that both Adversarial and Fourier Domain Adaptation methods effectively mitigate the domain gap, with Fourier Domain Adaptation offering competitive results with a lighter implementation.**

I. Introduction

With the rapid emergence and advancements in Deep Learning methods, Semantic Segmentation tasks have experienced substantial enhancements and improved results. Deep Learning refers to algorithms capable of emulating the operations of the human brain through artificial neural networks. Specifically, it is a machine learning technique based on neural networks, where tens or even hundreds of layers of neurons are stacked to enable greater complexity in rule establishment.

Semantic Segmentation, on the other hand, is a computer vision task that assigns a semantic class label to every pixel of an image using a deep learning algorithm (See Fig. 1). It stands as one of three sub-categories of image segmentation, alongside instance segmentation and panoptic segmentation.

Fig. 1: Images and their corresponding pixel-wise annotations

In this context, to train a model to classify every pixel of an image by assigning it a class like car, street, tree, etc.., a sufficiently large amount of images along with their ground-truth annotations is needed. This is costly in terms of human effort and time. For instance, fine-annotating a single image in the Cityscapes dataset took about 1.5 hours [1]. In contrast, synthetic datasets offer a more efficient alternative, as they can be annotated automatically without human intervention. For example, the GTA5 [2] synthetic dataset was annotated with pixel-level semantic segmentation ground truth for 25,000 images in just 49 hours. If the same technique used for Cityscapes were applied, it would have taken at least 12 person-years.

Although this approach solves a large problem and reduces the cost for Semantic Segmentation tasks significantly, it creates a domain gap between the source domain (dataset used for training) and the target domain (dataset used for testing) (See Fig. 2). The trained model is not able to generalize well, and its performance decreases when presented with images from the real-world, which are different than the ones used during training.

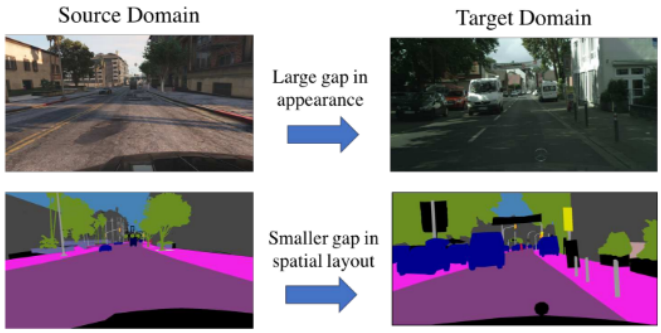


Fig. 2: Images from different domains share similar spatial layout and local context

We now proceed to describe our work. In a brief overview, We started by training a model to perform Semantic Segmentation on the *Cityscapes* dataset, then on the *GTA5* dataset. After that, we used the model trained on the *GTA5* dataset to annotate images from the *Cityscapes*. This allowed us to measure the domain-gap. We proceeded to try to bridge this gap by performing data augmentation on the training dataset, and observing how this affects the domain-gap. Then, we implemented an Adversarial Domain Adaptation (ADA) model that relies on a Generator-Discriminator structure to close the domain gap between the two domains. Again, we try to perform data augmentation during the training phase of the ADA model in an attempt to improve the performance. We measure and compare the performance of all our models using

Pixel Accuracy and Mean Intersection over Union (mIoU). Lastly, we explore possible extensions for our project to enhance performance, and to that aim we implement FDA, Fourier Domain Adaptation [3].

As for the structure of this paper, we dive deeper into both Semantic Segmentation and Domain Adaptation in the Related Works (II) section. In the Methodology (III) section, we explain from an algorithmic point of view the specific backbones, methods, and models used in our project. We give an overview of specifications and details of the datasets we employed in the Datasets section. Then, we describe the metrics used to compare models and measure their performance in the Performance Metrics (V) section. Finally, in the Experiments and Results (VI) section we report the trainings we performed and the results we obtained in each phase. We analyze them and make conclusions in the Conclusions (VII) section.

II. Related Works

A. Semantic Segmentation

Semantic segmentation assigns a category label to each pixel of an image, which is a fundamental but challenging task in computer vision research [4]. The emergence of Deep Learning has greatly promoted semantic segmentation research in latest years, as it dramatically increased the segmentation accuracy. It employs Fully-Convolutional Networks (FCN) [4], with which the learning process gradually bridges the inconsistency between high-level semantics and low-level features, making the network continually more aware of various semantic concepts. In the following, we group Deep Learning methods for Semantic Segmentation based on their supervision levels:

1) Supervised Methods: Fully supervised semantic segmentation methods rely on the availability of labeled training data, including original images and corresponding pixel-wise semantic annotations. These methods can be categorized as follows:

- **Context-based methods:** These methods leverage context information beyond pixel-level appearance to enhance semantic segmentation. They often employ dilated convolutions with flexible ”dilated rates” to increase the receptive field.
- **Feature enhancement-based methods:** They combine semantic-aware features from deep layers with spatial details from shallow layers to improve segmentation accuracy.
- **Deconvolution-based methods:** These methods use encoder-decoder architectures to gradually extract semantic features while upscaling low-resolution feature maps to high-resolution ones using unpooling and deconvolution operations.
- **RNN-based methods:** Addressing the limitation of modeling long-range dependencies, these methods utilize Recurrent Neural Networks (RNNs) to capture contextual information beyond adjacent pixels or patches.
- **GAN-based methods:** These methods employ adversarial training to enhance segmentation quality. The segmentation network partitions input images, while the adversarial network distinguishes between generator outputs and ground-truth labels, applying adversarial loss alternately to the generator and discriminator.
- **RGBD-based methods:** Utilizing RGBD images (combining RGB with depth information), these methods introduce an additional branch to extract depth features and employ multi-modal feature fusion to combine photometric and depth features for improved segmentation.
- **Real-time methods:** Despite achieving high accuracy, deep-learning-based semantic segmentation methods often have high computational costs. Real-time methods aim to enhance efficiency while maintaining segmentation accuracy. Examples include ENet, DIL, ESP, RefineNet, and BiSeNet.

2) Weakly-Supervised Methods: Weakly-supervised semantic segmentation methods operate without full annotations for training. Instead, annotations may include tags, scribbles, or bounding boxes, requiring less manual labeling effort compared to fully supervised methods. These methods can be categorized based on the type of annotation: tag-level supervision, scribble-level supervision, or bounding-box-level supervision.

3) Semi-Supervised Methods: In contrast to weakly-supervised methods, semi-supervised semantic segmentation assumes the availability of a small number of fully annotated training images, rather than a large number of weakly supervised ones. A typical scenario involves adapting a semantic segmentation model trained on a source domain to a target domain.

B. Domain Adaptation

Domain Adaptation is a machine learning technique that aims to close the gap between two domains that may have different data distributions or feature representations. It is a special case of *transfer learning* [5] whose goal is to uncover the common latent factors across the source and target domains and adapt them to reduce both the marginal and conditional mismatch in terms of the feature space between domains. From here, we define the *source domain* as the domain for which labeled data is available and used for training the model, and the *target domain* as the domain on which the model will be tested, and for which labeled data is either scarce or unavailable. Domain adpatation tasks can be grouped into three categories:

- Unsupervised: The basic setting is a transductive setting: it supposes that during training we have access to both the source domain and the unlabelled target domain. *The model has access to the target domain during training.*
- Semi-Supervised: It allows a relaxation with respect to unsupervised models, in particular, a small amount of labeled data is available in the target domain, along with a larger amount of unlabeled data.
- Fully-Supervised: Ample labeled data is available in the target domain, similar to traditional supervised learning but with a domain shift.

III. Methodology

A. Semantic Segmentation

*1) **BiSeNet: Network Architecture**:* BiSeNet, short for Bilateral Segmentation Network, is among the state-of-the-art methods proposed for semantic segmentation tasks. Such tasks require both rich spatial information and a sizeable receptive field. To elaborate, spatial information refers to details and

relationships present within spatial dimensions of an image, like edges, textures, shapes, and boundaries. While a receptive field is the region of the image that influences the output of a particular neuron, which tends to decrease in size as we move through the network. To address this dilemma, BiSeNet utilizes two paths: a **Spatial Path** with a small stride to preserve the spatial information and generate high-resolution features, and a **Context Path** with a fast down-sampling strategy to obtain sufficient receptive field. On top of the two paths, we introduce a new **Feature Fusion** module to combine features efficiently [6].

- **Spatial Path:** This path extracts the output feature maps that is 1/8 of the original image by stacking three convolution layers, retaining affluent spatial details. Each layer includes a convolution with stride = 2, followed by batch normalization and ReLU.
- **Context Path:** This path utilizes a lightweight model, like Xception, which can downsample the feature map fast to obtain large receptive field, encoding high level semantic context information. After obtaining feature maps with a large receptive field from the lightweight model, global average pooling is applied to summarize the entire feature map by computing the average of all its values, effectively capturing global context information. Then, a U-shape structure is used to fuse the feature maps obtained from the average pooling with the features obtained from the lightweight model.

In pursuit of better accuracy without loss of speed, a Feature Fusion Model FFM is proposed to fuse the Spatial and Context paths, and an Attention Refinement Model ARM is proposed to refine the final predictions.

- **Feature Fusion Model:** It combines output features from the Spatial and Context paths by concatenating them considering their varying levels of feature representation. Then it applies batch normalization, pools them into a feature vector, and computes a weight vector for feature re-weighting. This integration enhances performance by effectively blending spatial and contextual information.
- **Attention Refinement Model:** Global average pooling is employed to capture global context, then an attention vector is computed to guide feature learning. The attention mechanism's core idea is that the context vector should have access to all parts of the input instead of just the last one, and thus it enhances the model's performance by allowing it to focus on the most important parts of the input image.

The network architecture which consists of these four modules can be seen in Fig.3 [6].

*2) **STDC, BiSeNet Rethinked: Network Architecture**:* Although popular, BiSeNet faces challenges in real-time semantic segmentation:

- Reliance on lightweight backbones like Xception, borrowed from image classification tasks, may not be ideal for segmentation due to a lack of task-specific design.
- Imposed restrictions on input image size, potentially overlooking detailed appearance around boundaries and small objects.

The Short-Term Dense Concatenate Network (STDC) aims to address these issues and remove structure dependency by:

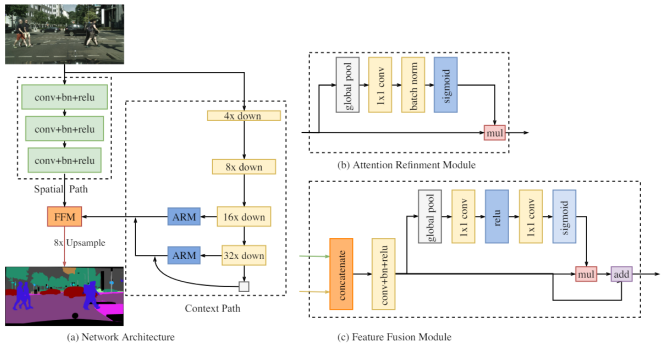


Fig. 3: (a) Network Architecture. The length of block indicates the spatial size, while the thickness represents the number of channels. (b) Components of the Attention Refinement Module (ARM). (c) Components of the Feature Fusion Module (FFM). The red line represents we take this process only when testing

- Gradual reduction of feature map dimensions and aggregation for image representation, forming the basic module of the STDC network.
- Proposal of a Detail Aggregation module in the decoder, integrating spatial information learning into low-level layers in a single-stream manner.
- Fusion of low-level and deep features to predict final segmentation results.

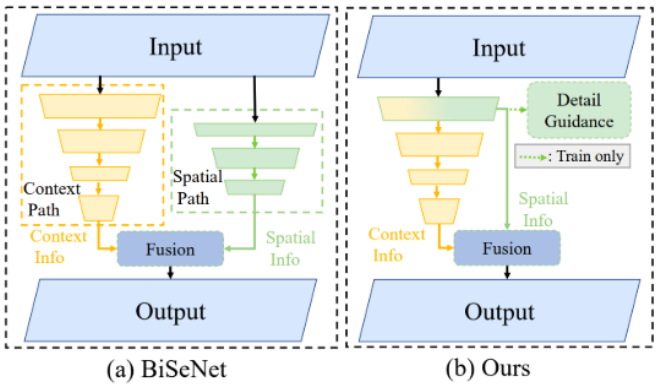


Fig. 4: (a) BiSeNet (b) STDC

- **Encoder:** The key component of the encoding network is the STDC module. It consists of several blocks denoted as $ConvX_i$, each contributing to the final output by sequentially processing the input features. The number of filters decreases geometrically across the blocks, except for the last layer which maintains consistency. Down-sampling occurs only in the second block to enrich feature information. Finally, the feature maps from all blocks are concatenated to form the final output of the STDC module. (See Fig.5) .The output of i-th block is calculated as follows:

$$x_i = ConvX_i(x_{i-1}, k_i) \tag{1}$$

- **Decoder:** The decoder in the network first processes the global average-pooled and down-sampled data from the encoder, utilizing the U-shape structure to upsample

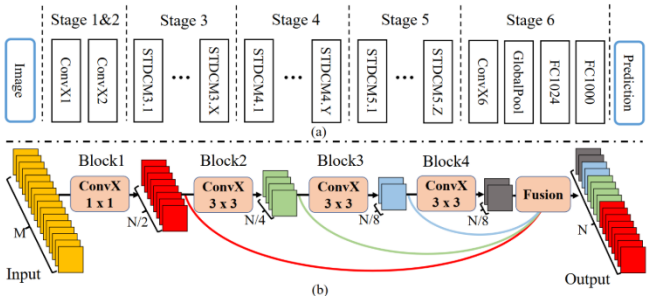


Fig. 5: (a) General STDC network architecture. ConvX operation refers to the Conv-BN-ReLU. (b) STDC module used in the network

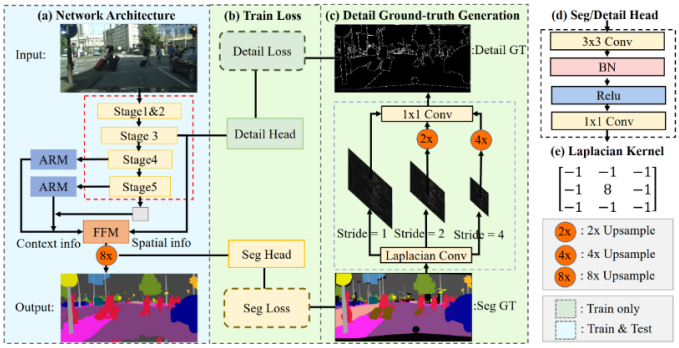


Fig. 6: Overview of the STDC Segmentation network.

these features and integrate them with corresponding features from later stages. It then refines the combined features using an Attention Refinement Module, enhancing the quality of the decoded features. Following this, a feature fusion operation merges the down-sampled feature map from Stage 3 with its counterpart from the decoder, effectively combining detailed information from the encoder with context information from the decoder. Finally, the segmentation prediction is generated through a series of operations including a 3×3 Conv-BN-ReLU layer followed by a 1×1 convolution. Overall, decoding operations primarily focus on upsampling, refining, combining features, and finalizing the segmentation output.

Detail Aggregation Module: To remove BiSeNet's structure redundancy, i.e, the Spatial Path, in the decoder, we introduce a detail aggregation module that integrates the learning of spatial information into low-level layers in a single-stream manner [7]. Since the spatial path can encode spatial details like boundaries and corners, we model the detail prediction as a binary segmentation task. To generate the detail ground-truth, a 2D convolution with a Laplacian kernel is applied using different strides. The feature maps are then upsampled and final labels are obtained by thresholding with 0.1. However, predicting the details is challenging due to class imbalance (fewer detail pixels compared to non-detail pixels). So a combination of binary cross-entropy loss and dice loss is used to jointly optimize detail learning. The dice loss measures the overlap between predicted maps and ground-truth:

$$L_{\text{dice}}(p_d, g_d) = 1 - \frac{2\sum_{i=1}^{H\times W} p_i d_i g_i d + \epsilon}{\sum_{i=1}^{H\times W}(p_i d_i)^2 + \sum_{i=1}^{H\times W}(g_i d_i)^2 + \epsilon} \quad (2)$$

So, the detail loss becomes:

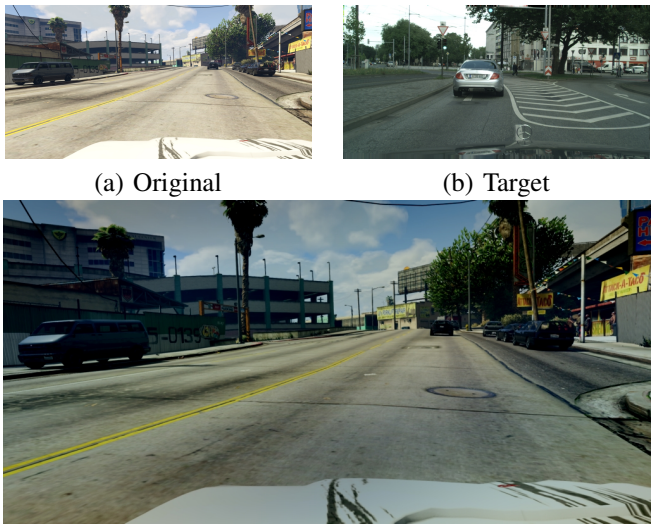
$$L_{\text{detail}}(p_d, g_d) = L_{\text{dice}}(p_d, g_d) + L_{\text{bce}}(p_d, g_d) \quad (3)$$

Finally, a detail head consisting of 3×3 Conv-BN-ReLU operator followed by a 1×1 convolution is used to produce the output detail map. This branch is discarded during inference, allowing the method to improve segmentation accuracy without additional inference cost.

An overview of the STDC network with all its modules can be seen in Fig.6

*3) **Data Augmentation, Extensions: FDA, SSL**:*

- **Data Augmentation:**Data augmentation techniques artificially generate different versions of an existing dataset to expand its size. By applying various transformations



(a) Original (b) Target

(c) Transformed

Fig. 7: Results after applying FDA to a sample image

or manipulations to the existing data while preserving the class labels (in supervised learning), these techniques create new artificial samples. This process introduces variations and perturbations into the data, which can enhance the robustness and generalization of machine learning models. Common example transformations include rotations, flips, scaling, cropping, and color adjustments. It is widely observed that using data augmentation leads to improved model accuracy, as it effectively increases the diversity of the training dataset without the need for additional real-world data

- **FDA:**
Fourier Domain Adaptation [3] is a technique for unsupervised domain adaptation that reduces the discrepancy between source and target distributions by swapping their low-frequency spectra. This approach is particularly useful in semantic segmentation, where densely annotated images are abundant in one domain (e.g., synthetic data) but scarce in another (e.g., real images). Unlike adversarial domain adaptation that rely on complex optimization to make the neural network invariant to domain selection, this method requires no training for domain alignment. Instead, it simply applies a Fourier Transform and its inverse. Despite its simplicity, this technique claims to achieve optimal performance when integrated into a standard semantic segmentation model. Generally, the segmentation network trained on D^s will have a

performance drop when tested on D^t. The technique requires a source dataset $D^s = \{(x_i^s, y_i^s)\}$ and a target dataset $D^t = \{(x_i^t, y_i^t)\}$.
Fourier Domain Adaptation can be formalized with the following:

$$x^{s\to t} = \mathcal{F}^{-1}\Big(M_\beta \cdot \mathcal{F}_A(x^t) + \\ + (1 - M_\beta) \cdot \mathcal{F}_A(x^s),\ \mathcal{F}_P(x^s)\Big) \tag{4}$$

where M_β is a mask whose value is zero except for the center region of size β that we want to transfer from the target to the source image, $\mathcal{F}_A$ and $\mathcal{F}_P$ respectively represents the amplitude and phase components of the Fourier's transformation. The loss function can be written as

$$\mathcal{L}_{fda}(\phi^w; D^{s\to t}, D^t) = \mathcal{L}_{ce}(\phi^w; D^{s\to t}) \\ + \lambda_{ent}\mathcal{L}_{ent}(\phi^w; D^t) \tag{5}$$

where $\lambda_{ent}\mathcal{L}_{ent}(\phi^w; D^t)$ is used as a penalty for the decision boundary to cross clusters in the unlabeled space, in order to have smoother boundaries. In Fig. 7 we can see an example of an image transformation, given the original and target images.

- **SSL:** Self-Supervised Learning (or, Self-Supervised training) is a common way of attempting to boost the performance by using highly confident pseudo-label predicted with another model as if they were ground truth. Without regularization, the task would be self-referential so a regularization technique is needed. The mean of the predictions of different models is used to regularize self-learning. The different models $\phi^w_{\beta_m}$ are trained from scratch with different β_m. It is also applied a threshold on the confidence values of each prediction. More specifically, for each semantic class, the predictions with confidence that is within the top 66% or above 0.9 is accepted, otherwise it is considered as *ignored class*. The loss then can be seen as (5) with in addition a cross-entropy loss on the D^t using the pseudo labels, $\hat{D}^t$.

$$\mathcal{L}_{sst}(\phi^w; D^{s\to t}, D^t) = \mathcal{L}_{fda}(\phi^w; D^{s\to t}, D^t) \\ + \mathcal{L}_{ce}(\phi^w; \hat{D}^t) \tag{6}$$

An example can be visualized in Fig. 8

B. *Adversarial Domain Adaptation*

For task-specific applications like semantic segmentation, the property that pixel-level predictions are structured outputs that contain information spatially and locally is exploited to propose a Domain Adaptation model based on adversarial learning in the output space. Our domain adaptation algorithm consists of a segmentation network G and a discriminator D. The network is trained adversarially to make segmentation predictions for source images (P_s) and target images (P_t) similar, using the discriminator to differentiate between source and target domains, thus encouraging G to generate consistent segmentation distributions across both domains [5]. The objective function for the network consists of the two loss functions from both modules:

$$L(I_s, I_t) = L^{seg}(I_s) + \lambda_{adv}L_{adv}(I_t) \tag{7}$$

For a full algorithmic overview, see Fig.9

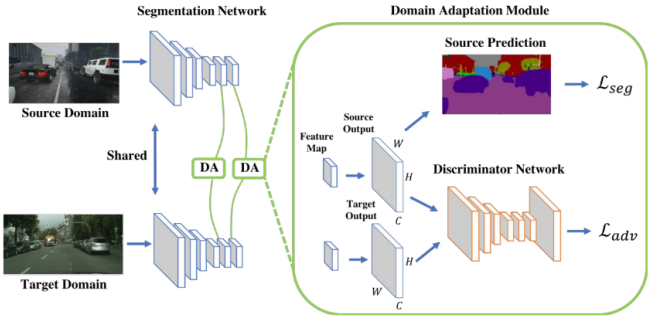


Fig. 9: Overview of the Adversarial Domain Adaptation Network.

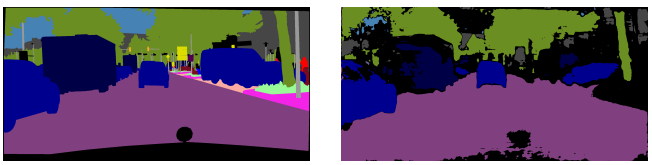
*1) **Segmentation Network**:* For the network architecture, we adopt the DeepLab-v2 framework with a ResNet-101 backbone pre-trained on ImageNet, modify strides for higher resolution, use dilated convolutions to enlarge the receptive field, employ Atrous Spatial Pyramid Pooling (ASPP) as the classifier, and up-sample the softmax output to match the input image size for high-quality segmentation results.As for the segmentation loss, it is defined as the cross-entropy loss for i mages from the source domain. Images from the target domain are forwarded to G. To make the distribution of P_t closer to P_s, we use an adversarial loss. This loss is designed to train the segmentation network to fool the discriminator by maximizing the probability of the target prediction being considered as the source prediction.

*2) **Discriminator** D:* The discriminator's task is to distinguish between samples from the source and target domains using a cross-entropy loss:

$$L_d(P) = -\sum_{h,w}(1-z)\log\Big(D(P)(h,w,0)\Big) \\ + z\log\Big(D(P)(h,w,1)\Big) \tag{8}$$

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

It is fed the segmentation softmax output P.The discriminator architecture is designed with fully-convolutional layers to retain spatial information. It consists of 5 convolutional layers with a 4×4 kernel and a stride of 2. The channel configuration is 64, 128, 256, 512, 1, respectively. Each convolution layer, except the last one, is followed by a leaky ReLU activation with a parameter of 0.2. An up-sampling layer is added to the final convolutional layer to match the output size to the input. Batch normalization layers are excluded to facilitate joint training with the segmentation network using a small batch size.



(a) Ground Truth (b) Pseudo Label

Fig. 8: Pseudo Label

IV. Datasets

A. Cityscapes

The Cityscapes dataset is a large-scale benchmark dataset used for semantic urban scene understanding. It contains 5,000 high-quality pixel-level annotated images, divided into 2,975 images for training, 500 for validation, and 1,525 for testing. They have a resolution of (1024,2048), and the labelling was obtained through a meticulous and detailed manual annotation process, requiring around 1.5h per image. [1]. The dataset features 19 semantic classes organized into several categories: flat, construction, object, nature, human, and vehicle. Captured from 50 different cities under various weather conditions and seasons, it is diverse and contains challenging scenarios. We use it in our project as a training and validation dataset in various experiments, but ultimately, it is our *chosen target dataset.*

B. GTA5

The GTA5 dataset is a synthetic dataset widely used for semantic segmentation in urban environments. It contains 24,966 high-resolution images extracted from the video game Grand Theft Auto V, annotated with pixel-level labels. These images are automatically labeled using the game engine, which generates ground-truth annotations by leveraging the internal representations and object information available within the game. the images have a resolution of (1052, 1914) and the entire labelling process took only 49 hours. [2]. The dataset features the same 19 semantic classes as in the Cityscapes dataset. We use it in our project as a training and validation dataset in various experiments, but ultimately, it is our *chosen synthetic source dataset.*

V. Performance Metrics

In our project, we use the following two performance metrics to evaluate the model.

A. Pixel Precision

The pixel precision simply reports the percent of pixels in the image which were correctly classified. It is also computed both for each class separately, and globally across all classes. When evaluating per-class pixel accuracy, we are essentially assessing a binary mask. A true positive occurs when a pixel is correctly predicted to belong to the specified class (as indicated by the target mask), while a true negative occurs when a pixel is accurately identified as not belonging to that class.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{9}$$

This metric however, can provide misleading results when the class representation is small within the image, as the measure will be biased in mainly reporting how well you identify negative case (ie. where the class is not present).

B. Mean Intersection over Union

The Intersection over Union (IoU) metric allows us to quantify the percent overlap between the target mask and our prediction output. It is also very closely related to the Dice coefficient which is used as a loss function during training in the Detail Aggregation Module. The IoU measures the number of pixels common between the target and prediction masks divided by the total number of pixels present across both masks.

$$IoU = \frac{target \cap prediction}{target \cup prediction} \tag{10}$$

The IoU score is calculated for each class separately and then averaged over all classes to provide a global, mean IoU score of our semantic segmentation prediction.

VI. Results and Experiments

A. Model and Training Setup

We employed the BiSeNet model with a STDC network pretrained on ImageNet for our experiments. The training process was conducted over 50 epochs with a polymonial decaying learning rate, starting from 0.01. We investigated the effects of varying the batch size and the choice of optimizer on the model's performance. Specifically, we experimented with batch sizes of 4 and 6 and used three different optimizers: RSM, Adam, and SGD.

B. Data Preparation

Two datasets were utilized for training and evaluation: Cityscapes and GTA. The datasets underwent the following preprocessing transformations:

- **Cityscapes**:
  - Resize to 512×1024 pixels
  - Normalization
- **GTA**:
  - Resize to 720×1280 pixels
  - Normalizion
- **GTA + Augmentation**:
  - Random Crop to 720×1080
  - Random Color/Brightness/Hue tweaks
  - Random geometric transformation (e.g., horizontal flip)

For validation, the images were resized to a resolution of 512×1024 and normalized, without applying any additional transformations.

C. Training on Real-World dataset: Cityscapes

We first trained our model on the Cityscapes dataset and validated it on the same dataset to establish a baseline for comparison. The table below *(see Table I)* summarizes the performance of the model under different optimizer and batch size configurations
In general, it can be observed that the Stochastic Gradient Descent (SGD) optimizer tends to outperform the other optimizers across different batch sizes in terms of both mean Intersection over Union (mIoU) and precision.

D. Training on Synthetic dataset: GTA5

We then trained our model on the synthetic GTA5 dataset and validated it on the same dataset to establish a benchmark for comparison. Notably, we found that *(see Table II)* the Stochastic Gradient Descent (SGD) optimizer, along with the same configuration of batch size 6, consistently yielded the most favorable results in terms of both mean Intersection over Union (mIoU) and precision.

TABLE I: Training on Cityscapes ↪ Validation on Cityscapes

Optimizer	Batch Size	mIoU (%)	Precision (%)	Training Time (avg s/epoch)
RMS	4	42.69	77.95	74.30
	6	42.82	78.16	71.08
ADAM	4	46.51	77.68	76.76
	6	45.78	78.98	71.16
SGD	4	52.10	80.21	74.56
	6	**53.90**	**80.34**	70.00

Results of the model trained on Cityscapes, resized to $(512, 1024)$ and normalized. Validated on Cityscapes with the same transformations.

TABLE II: Training on GTA5 ↪ Validation on GTA5

Optimizer	Batch Size	mIoU (%)	Precision (%)	Training Time (avg s/epoch)
RMS	4	42.15	76.91	145.16
	6	41.99	77.22	140.26
ADAM	4	42.55	76.90	150.00
	6	42.38	77.39	142.46
SGD	4	51.46	79.13	141.54
	6	**55.34**	**79.50**	142.46

Results of the model trained on GTA5, resized to $(720, 1280)$ and normalized. Validated on GTA5 with the same transformations.

E. *Evaluating the Domain Shift*

We now delve into evaluating the domain shift in semantic segmentation by training our model on a synthetic dataset, specifically GTA, and evaluating its performance on the real-world dataset, Cityscapes. In particular, we re-use the model trained on GTA5 in the previous part, and we perform validation on the real-world dataset Cityscapes. Additionally, we explore the potential benefits of augmenting the GTA5 dataset and we assess its impact on model performance. Through this investigation, we aim to gain insights into the generalization capabilities of our model across domains and examine the effectiveness of data augmentation techniques in bridging the domain gap between synthetic and real-world datasets.
Note: from this point onward, we focus exclusively on the Stochastic Gradient Descent (SGD) optimizer, given its consistent superiority in previous experiments.

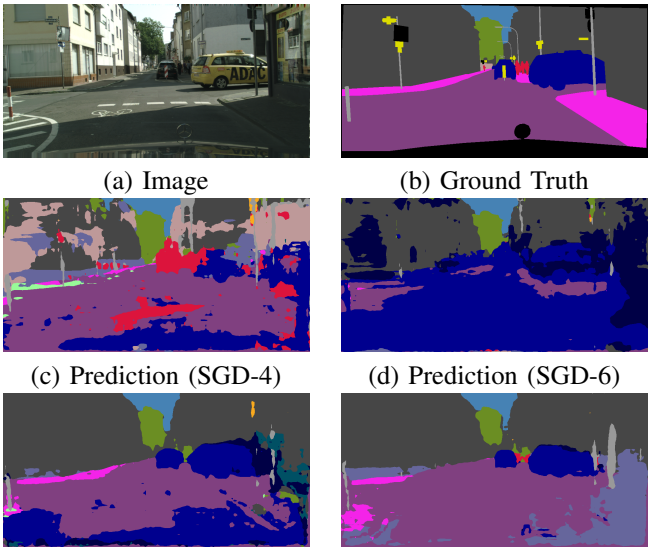
Observing the results on Table III, Table IV and Fig. 10 we can assert that augmenting the GTA5 dataset led to enhanced precision and mean Intersection over Union (mIoU) scores for both batch sizes 4 and 6 when evaluated on the Cityscapes dataset. Compared to the non-augmented results, the augmented dataset exhibited higher precision, indicating better pixel-wise classification accuracy, and increased mIoU values, signifying improved overall segmentation performance. These findings underscore the efficacy of data augmentation techniques in enhancing the generalization capabilities of semantic segmentation models across different domains.

TABLE III: Evaluation of Domain Shift from GTA to Cityscapes

Optimizer	Batch Size	Precision (%)	mIoU (%)
SGD	4	44.34	14.37
SGD	6	38.67	14.43

TABLE IV: Evaluation of Domain Shift with augmentation from GTA to Cityscapes

Optimizer	Batch Size	Precision (%)	mIoU (%)	Training Time (avg s/epoch)
SGD	4	69.23	24.47	219.22
SGD	6	67.82	24.76	197.36



(a) Image (b) Ground Truth (c) Prediction (SGD-4) (d) Prediction (SGD-6) (e) Prediction (Aug-SGD-4) (f) Prediction (Aug-SGD-6)

Fig. 10: Comparison of segmentation results with and without augmentation.

F. *Unsupervised Adversarial Domain Adaptation*

We then moved on to adversarial domain adaptation, a technique categorized under unsupervised domain adaptation. This approach is classified as "unsupervised" because it does not require labeled data from the target domain for training. Instead, it leverages a discriminator model to differentiate between the source and target domains, thereby reducing the domain shift. This technique, as detailed by the loss function in equation (7), achieved better results (Table V) than the baseline (Table III, IV). Specifically, we achieve a significantly higher mIoU than we do using the baseline (Table IV). The improved performance demonstrates that adversarial domain adaptation is effective in enhancing the task, further validating its utility in addressing domain shift issues.

TABLE V: Adversarial Domain Adaptation Evaluation

Augmentation	Batch Size	Precision (%)	mIoU (%)	Training Time (avg s/epoch)
No	4	69.67	29.19	135.32
	6	64.42	26.25	138.58
Yes	4	72.94	30.11	146.64
	6	73.03	30.91	141.92

G. *Improvements*

We are now exploring the improvements achieved through the application of Fourier Domain Adaptation (FDA) [3]. FDA offers significant advantages in reducing domain shift by transforming synthetic images to resemble the target domain.

This transformation process mitigates the disparity between the training and target datasets, thereby enhancing the model's performance on real-world data. One of the key benefits of FDA is its efficiency; it does not require complex or computationally intensive networks. Instead, a simple Fast Fourier Transform and its inverse suffice to perform the necessary domain adaptation. This approach not only accelerates the training process but also leverages the inherent properties of Fourier transforms to achieve better generalization across domains. In this experiment, we trained different models using the SGD optimizer while varying the batch size and the β parameter. Specifically, we used batch sizes of 4 and 6, and $\beta \in \{0.01, 0.03, 0.05\}$. To further enhance the model's performance, we applied the entropy loss only after the first 20 epochs. Additionally, we experimented with and without augmentation techniques on the GTA (source) dataset to assess their impact on the domain adaptation process.

TABLE VI: Fourier Domain Adaptation Evaluation

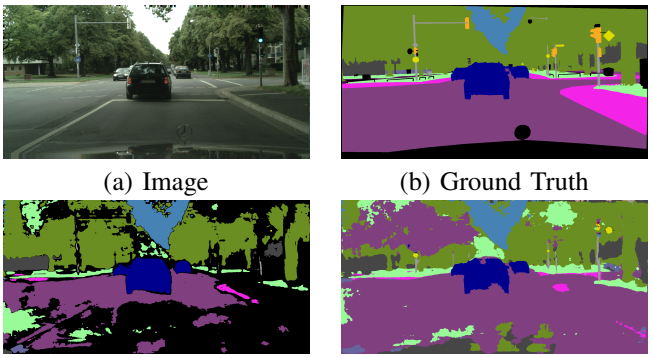
Augmentation	Batch Size	β	Precision (%)	mIoU (%)	Training Time (avg $s/epoch$)
No	4	0.01	71.75	30.37	158.4
		0.03	72.49	31.32	159.64
		0.05	72.44	31.44	141.02
	6	0.01 [b]	74.14	33.71	145.50
		0.03	73.32	32.93	143.46
		0.05	73.16	32.37	159.66
Yes	4	0.01	72.13	31.12	171.34
		0.03	72.59	31.77	173.50
		0.05	71.48	30.63	171.03
	6	0.01 [a] [b]	**74.31**	**34.30**	158.24
		0.03 [b]	73.50	33.96	158.24
		0.05	73.34	32.99	159.66

[a] Best Model
[b] Best Models with lower β, used for FDA-SST

Using the best models identified so far with FDA, we generated pseudo-labels for the Cityscapes [1] dataset to be able to perform Self-Supervised Training. We selected the top three models with low β values (Table VI [(b)]), as these impose less bias due to their closer alignment with the target dataset and smaller variance [3]. This approach aims to leverage the strengths of FDA while minimizing potential discrepancies between the synthetic and real-world domains. With the newly generated pseudo-labels, we then trained a new model from scratch, limiting the training to the augmented dataset, using the FDA-SST Loss (6), and the results can be seen at Table VII and Fig 11

TABLE VII: Fourier Domain Adaptation with Self-Supervised Training Evaluation

Batch Size	β	Precision (%)	mIoU (%)	Training Time (avg $s/epoch$)
4	0.01	74.20	32.01	181.34
	0.03	74.06	31.96	180.9
	0.05	74.02	32.13	180.36
6	**0.01**	**74.53**	**33.62**	**176.88**
	0.03	74.16	32.99	189.36
	0.05	74.26	31.95	203.06



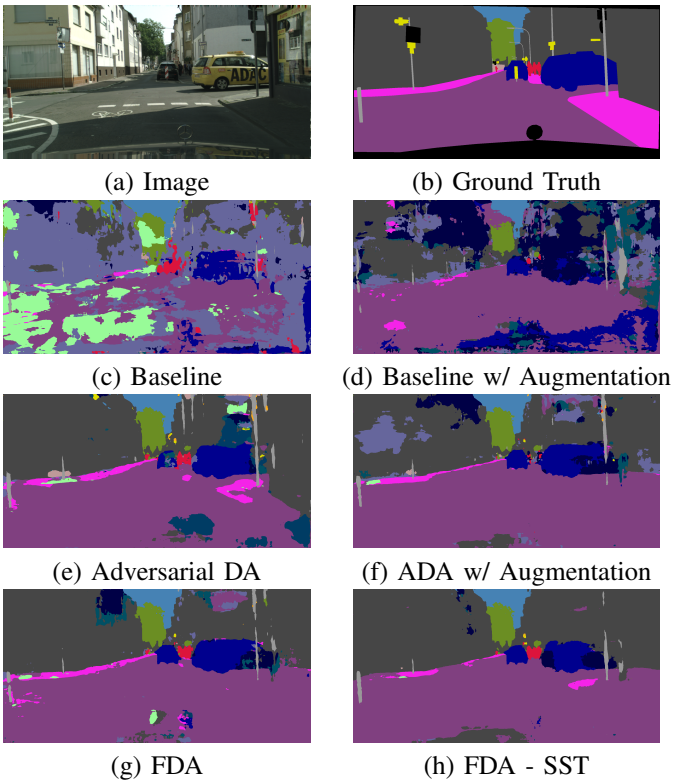
(a) Image (b) Ground Truth

(c) Pseudo Label (d) Prediction

Fig. 11: FDA SST - Visualization

VII. Conclusions

Among the techniques explored, data augmentation is the easiest to implement and proves to be effective, increasing our mIoU from 14% to 24%. Combining it with ADA, our network performed even better considering both precision and mean Intersection over Union compared to the baseline (ca. +16% mIoU compared to our baseline), but the combination of FDA and data augmentation has shown the best results in addressing domain adaptation(ca. +20% mIoU compared to our baseline). Its ability to transform synthetic images to resemble the target domain, its computational simplicity, integration with self-supervised training (SST), combined with data augmentation make it an effective approach to improve the performance of semantic segmentation models on real-world data. In particular, the use of FDA models with low β values to generate pseudo-labels for SST has shown promise to further improve performance. The code developed for this project can be found on GitHub



(a) Image (b) Ground Truth

(c) Baseline (d) Baseline w/ Augmentation

(e) Adversarial DA (f) ADA w/ Augmentation

(g) FDA (h) FDA - SST

Fig. 12: Models Comparison

REFERENCES

[1] 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." https://doi.org/10.48550/arXiv.1604.01685, 2016.

[2] S. R. Richter, V. Vineet, S. Roth, and V. Koltun, "Playing for data: Ground truth from computer games." https://doi.org/10.48550/arXiv.1608.02192, 2016.

[3] Y. Yang and S. Soatto, "Fda: Fourier domain adaptation for semantic segmentation." https://doi.org/10.48550/arXiv.2004.05498, 2020.

[4] S. Hao, Y. Zhou, and Y. Guo, "A brief survey on semantic segmentation with deep learning." https://www.sciencedirect.com/science/article/pii/S0925231220305476, 2020.

[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." https://doi.org/10.48550/arXiv.1802.10349, 2020.

[6] C. Yu, J. Wang, C. Peng, C. Gao, G. Yu, and N. Sang, "Bisenet: Bilateral segmentation network for real-time semantic segmentation." https://doi.org/10.48550/arXiv.1808.00897, 2018.

[7] M. Fan, S. Lai, J. Huang, X. Wei, Z. Chai, J. Luo, and X. Wei, "Rethinking bisenet for real-time semantic segmentation." https://doi.org/10.48550/arXiv.2104.13188, 2021.