Research Internship Report

Subject : Adapting weather perception models for autonomous off-road driving

GOLEBIEWSKI ADRIEN

Internship carried out from 10/04/2023 to 06/10/2023

Internship tutors : M. Howard Mahé

Éducational institution : Université Paris-Dauphine PSL / Master 2 IASD

Internship host company : Safran TECH - Rue des jeunes Bois, 78117 Châteaufort

Acknowledgments

Firstly, I'd like to thank my internship tutor Howard Mahé for his confidence and optimism, and for mentoring me throughout these 6 months. I'd like to thank him for his sound advice, which helped fuel my thinking.

His role as facilitator and his advice played a decisive role in the smooth running and success of the internship.

I would also like to thank Camille Chapdelaine and Ahmed Benaichouche, who took over from my internship tutor during his periods of absence. Their complementary vision and expertise were also invaluable.

In general, I would like to thank all my colleagues in the Autonomous Vehicle Laboratory, and more particularly the CASPer team, who encouraged me during my 6-month internship and enabled me to advance my projects under excellent conditions.

I pay my gratitude to Safran Group, and especially Safran TECH. It is a great and unique pleasure to me that I have got a chance to work in this research environment.

Finally, I would like to thank the Université Paris Dauphine PSL and all the teaching staff of the IASD Master 2 program for their commitment to the course and their support during this year.

Abstract

Nowadays, aerospace, automotive and other industrial manufacturers are leading some works to carry out experiments, provide expertise and monitor research projects related to vehicle autonomy. The key to autonomous vehicles is their ability to make decisions on their own, whatever the situation. In addition to trajectory planning and associated control methods, compiter vision/perception bricks are also essential for understanding the environment. For autonomous driving, perception algorithms are mainly based on Artificial Intelligence and more particularly on neural network models. Therefore they require large amounts of data to be trained correctly.

For semantic segmentation, the annotation phase is costly, as each pixel must be labelled, and extremely tedious and time-consuming. However, the deployment of semantic segmentation solutions for autonomous driving requires the algorithm to be robust to deviations in the distribution of real field data, in order to guarantee safety. Indeed, it is customary to explain that some driving situations are more complex than others, and that the situations to be taken into account are highly variable
Directly generalizing the models trained on one large-scale labeled source domain to another related and unlabeled target domain could be a solution but usually may not perform well because of the domain shift.
Therefore, transferring the labeled data in the source domain to the target domain thanks to the Domain Adaptation solution is a promising alternative that we'll be exploring in this report.
Unsupervised domain adaptation eliminates the need to annotate the target domain. It has been successfully applied to many uses cases like Clear weather to Adverse Weather, Synthetic to Real, Daytime to Nighttime or Location A to Location B cases.

These 6 months of internships were an opportunity to focus on the typical case of Clear weather to Adverse Weather with experimentations on public data and consider whether these experiments can be transposed to applications using a private Safran dataset.

Key words: autonomous driving, perception, neural networks, segmentation, domain adaptation, adverse weather

Contents

1 Introduction

High-tech aerospace is the most important business sector for the Safran Group, one of the world's leading aircraft equipment suppliers. As a world's aircraft manufacturer, the Safran Group, and more specifically its Safran Tech research center, is faced with rapid technological change, which it must keep pace with by developing the most innovative technologies and equipment to best meet the needs of its customers. The latter are increasingly interested in system autonomy to define a more flexible mobility of the future without any human intervention. It can learn and adapt to dynamic environments, evolving as the surrounding environment changes, and can therefore be applied to any industrial or operational problem.

In the context of technologies developed for autonomous driving of military vehicles, the existing datasets for semantic semantic segmentation are captured either in off-road environments under nominal conditions, or in harsh weather conditions in urban environments. For autonomous driving, perception algorithms are mainly based on neural network models, and therefore require large amounts of correctly trained data. For semantic segmentation, the annotation phase is costly, as each pixel must be labelled, as well as extremely time-consuming. The deployment of semantic segmentation solutions for autonomous driving requires the algorithm to be robust to deviations in the distribution of real field data, in order to guarantee safety. In this context, the internship will focus on the development of methods for unsupervised domain adaptation of semantic segmentation algorithms to adverse weather conditions (fog, rain, snow). Unsupervised domain adaptation eliminates the need to annotate the target domain, in this case images in harsh weather conditions.

Recent methods have drastically reduced the gap between adapted and supervised models, with Transformers in particular proving to be excellent candidates for adaptation. In addition, adaptation to adverse domains makes it possible to regularize the model for nominal conditions. The methodology for evaluating adaptation methods and analyzing performance will draw on state-of-the-art benchmarks.

The report will follow the internship structure and will be divided into four parts. Firstly, I will present in general the functioning of Safran Group and more globally the professionnal and research context of my internship.
Next, I will describe the problem solving of the internship problem with on the one hand, a summary of the state of the art in which we will find detailed information about the different Domain Adaptation (DA) methods that have been developped until today, and on the other and, the techniques and methods choosen within this state-of-the-art environment.
Then, various results and experiments will be presented from a qualitative and quantitative point of view in a section dedicated to the study conducted on synthetic data and then evaluated on a Safran benchmark. Precise interpretations will then be given to highlight the research improvements and define the limitations and areas for progress of the solutions developed.

2 Professional Context

This first part of the report is devoted to a brief presentation of the Safran Group and Safran Tech in general.

2.1 Safran Group

Safran is an international high-technology group, present in 27 different countries with over 76,000 employees and sales of 15.3 billion euros in 2021. Its main target market is the aerospace sector, both civil and military. But also the automotive and rail markets, thanks to its expertise in electrical systems. Today, Safran is recognized as a leading equipment manufacturer in the field of aerospace propulsion, aircraft systems, aircraft interiors and defense.

Fig. 1: Take-off of an A320

Fig. 2: In May 2023, Safran Nacelles delivered the first LEAP-1A propulsion systems to Airbus for the A320.

Its organization is based on a parent company, Safran SA, which manages the Group's management and development, as well as its legal, tax and financial affairs. The various subsidiaries, for their part, are organized by business line, and manage the operational aspects of their respective activities, as well as internal control. This organization is illustrated in the diagram below (figure 1-2).

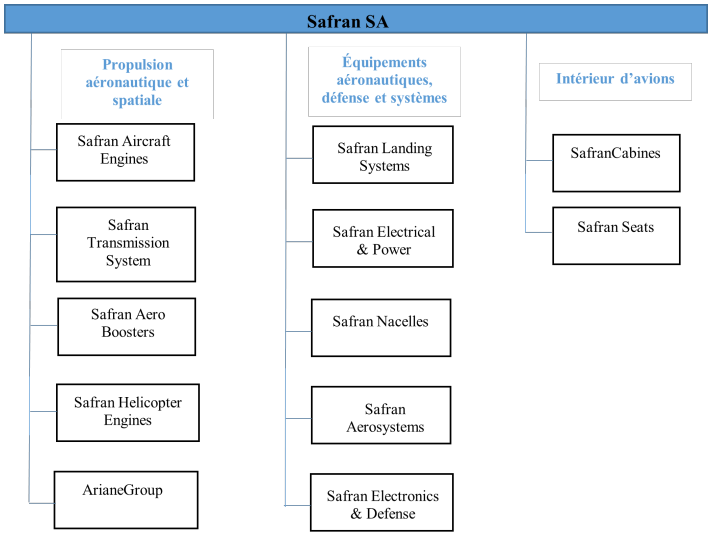

Figure 1-2. Schéma d'organisation succinct du groupe Safran

Fig. 3: Safran group organization chart

2.2 Safran TECH and the DST department

Safran's two main strategic priorities are to decarbonize its products and operations, and to strengthen its role in sovereignty activities. This is achieved by stepping up research and technology efforts to meet the needs of our customers and face up to the climate challenge. To this end, Safran Tech, was inaugurated in 2015 and now accounts for a quarter of the Group's Research and Technolgy efforts. It includes almost 500 researchers of over 20 different nationalities with almost 1000 patents and more than 150 scientific publication, divided into three divisions: materials and processes, energy and propulsion, and digital science and technologies. Its research program is drawn up jointly with the Group's companies in order to establish a bond of trust that enables them to seek out skills and value.

The internship was carried out in Safran Tech's Digital Science and Technologies (DST) department. It is the department that is charged with developing innovative image analysis tools that meet the needs of efficiency and robustness required by the aerial problems studied in Safran while inspiring from already existing machine/deep learning algorithms. This department is diversified and complete. It gathers different teams, each one working on a particular domain within the use of Machine Learning (ML)/ Deep Learning (DL). For instance, there are some who are specialized in perception problems (aerial but also terrestrial), while others are specialized in Non Destructive Inspection System (NDIS), or even in data analysis for predictive maintenance.

3 Internship Context

3.1 CASPer team activity

This internship was carried out in Safran Tech's Digital Science and Technologies (DST) department, and more specifically within the CASPer unit (Trustworthy AI, Simulation, Perception). The CASPer unit conducts research on a wide range of topics, from artificial intelligence for the control and monitoring of complex systems, to autonomous vehicles and systems.
Within Safran Tech, the CASPer team is part of the joint laboratory set up by Safran, Valeo and the PSA Group to accelerate the joint development of driver assistance solutions and autonomous vehicles.

3.2 Internship Problem Statement

In order to better understand the scope of the internship, in the following chapter we will address the problem of exploiting the unsupervised domain adaptation framework to solve the problem of changing weather conditions for segmentation tasks on Safran data.

UDA can be applied to many uses cases like Clear weather to Adverse Weather, Synthetic to Real, Daytime to Nighttime or Location A to Location B cases. For example, about the last case, we have below the results of a segmentation model trained on the Cityscapes Dataset (left image) and tested on the San Francisco Dashcam data set (right dataset).

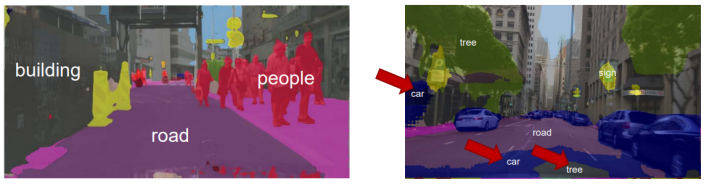

Fig. 4: An example of dataset bias or domain shift. The regions pointed out by red arrows are segmented with incorrect class labels. [1]

We can see that directly generalizing the models trained on one large-scale labeled source domain to another related and unlabeled target domain usually may not perform well because of the domain shift. We can make the same observations for the others above-mentioned use cases. Therefore, transferring the labeled data in the source domain to the target domain thanks to the Domain Adaptation solution is a promising alternative that we'll be exploring in this report.
On a regular basis, Safran Tech's CASPer team has been acquiring data from autonomous vehicle acquisition tests. The data has a very small number of classes and was acquired under nominal, non-adverse weather conditions. One possibility is to acquire additional datasets under adverse weather conditions and annotate the data. But this comes at a considerable cost.

The aim of the internship is to confirm that UDA can be a solution for task segmentation problems in adverse weather conditions applied to Safran dataset. To provide an answer to this problem, the internship will focus on the RUGD synthetic data and on all the segmentation and UDA tests carried out on the latter as a potential proxy.

To evaluate this, a barometer has been defined whose objective is to provide indicators of the performance of a domain adaptation experiment. The limits of the barometer are on the one hand the performance of the source-only model tested on the target dataset, and on the other hand the performance of an oracle model which would have had access to the train set of the target dataset. Ideally, the good performance of a model learned by domain adaptation should be halfway between these 2 limits.

In order to take account of adverse weather conditions in the experiments, data augmentation issues have also been addressed and questioned. Finally, we wanted to find out whether RUGD could be a good proxy for evaluating the performance of domain adaptation for Safran private data.

3.3 Related previous work

Within the CasPer team, no work on domain adaptation has yet been carried out, either on Safran data or on open source data, before to the start of the internship. On the other hand, Safran private dataset, named "Samba" was collected and has already been used several times by the Casper team. To put this internship topic into perspective, it would be interesting to understand how and why the CasPer team has used its data to date.

Indeed, a project has been launched by Valeo, in partnership with Safran, TwinsWheel and INRIA. Its aim is to design active safety solutions for autonomous mobility that are affordable and can be deployed rapidly. One of the technological solutions to be developed in this project concerns autonomous logistics: robots and droids for first- and last-mile delivery, as well as assistance to healthcare personnel and vulnerable people following natural disasters or pandemic situations. For this use case, Safran's target robot is a 4X4 military vehicle equipped with various sensors for autonomous functions.

To achieve this task, the vehicle must be able to move autonomously in unknown (unmapped) environments, which may or may not be structured (off-road, forests, etc.). Autonomous navigation requires the vehicle to perceive and understand the environment both geometrically and semantically, in order to move optimally in navigable areas while avoiding obstacles. It is therefore necessary to generate navigability maps around the vehicle, which are updated at defined time intervals (depending on vehicle speed and dynamic objects in the environment).

As the proposed solution is based on a deep learning approach, it is necessary to build up a fairly large database that is representative of the technology's target operating domain. SafranTech's vehicle is an indispensable tool for this. A fuller description of the Samba database can be found in the appendix A.1.

3.4 Process and timeline

After a familiarization phase, the research project began in early April with a bibliography phase and an understanding of the state of the art in UDA. The aim was first and foremost to understand the challenges of this method, to understand how the various solutions differ from one another, and to draft an initial deliverable for Safran Tech detailing all this information.

At the beginning of June, the first phase of development began with preliminary experiments, enabling us to get to grips with the development framework and neural network architectures, and to obtain initial interpretations. Finally, the technical phase culminated in a final development phase, which responded to our initial problem and brought new research results.

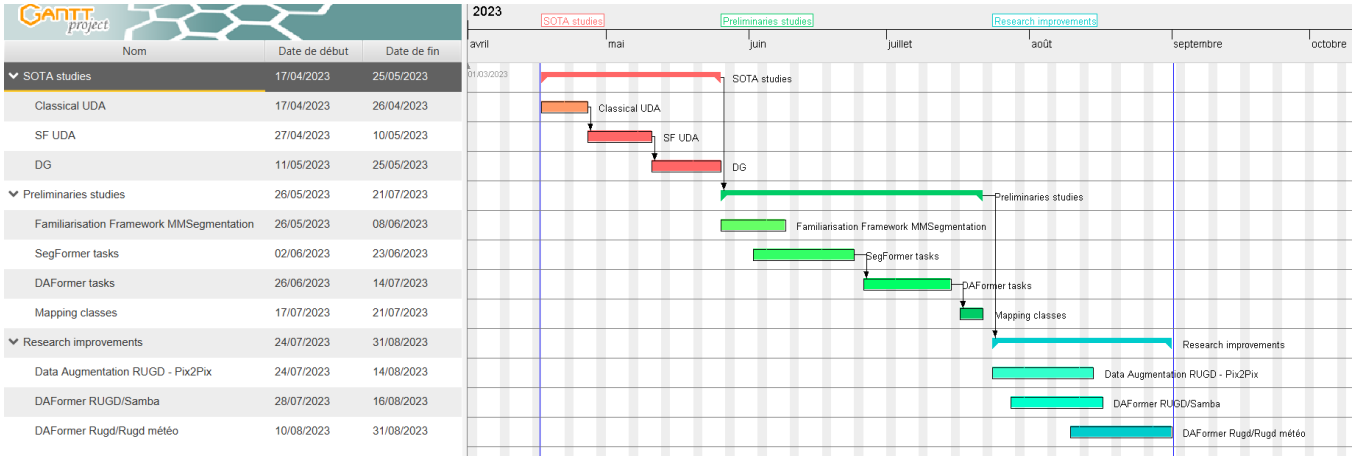

Fig. 5: Gantt Research project of the internship

The GANTT schedule above ends on September 1, the date on which I submitted this report. I have indeed succeeded in obtaining some initial research results and in responding to the problematic set beforehand. However, the results obtained and the associated interpretations can still be deepened by further experimentation. These lines of enquiry will be addressed later in the report, before the official end of the internship.

4 Problem solving proposal

The main goal of this section is to present the different strategies and methods that were used in order to solve the problem explained in the previous paragraphs. For this, it is important to start with a state of the art of the different methods in the Domain Adaptation field. Then, the main methods that we chose to implement are explained as well as the reason behind these choices. Finally, present in details the functioning and the characteristics of our technical and research work environment.

4.1 State of the art

4.1.1 Transfer learning

Before even mentioning the term Domain Adaptation, it is strongly needed to talk first about a more generic and well-known concept in the Deep Learning field, which is Transfer Learning (TL). TL is basically the ability to exploit the knowledge learned by a machine learning model from a source domain/task and apply it to some target domains/tasks, that, somehow, share some similarities with it.

Transfer learning is usually done for tasks where your dataset has too little data to train a full-scale model from scratch.
The most common incarnation of transfer learning in the context of deep learning is the following workflow:

- Take layers from a previously trained model.
- Freeze them, so as to avoid destroying any of the information they contain during future training rounds.
- Add some new, trainable layers on top of the frozen layers. They will learn to turn the old features into predictions on a new dataset.
- Train the new layers on your dataset.

A last, optional step, is fine-tuning, which consists of unfreezing the entire model you obtained above (or part of it), and re-training it on the new data with a smaller learning rate wrt. the ones used for new layers. This can potentially achieve meaningful improvements, by incrementally adapting the pretrained features to the new data. More explicitly, in the classic supervised learning scenario, if we intend to train a model for some task and domain, we assume that we are provided with labeled data for the same task and domain in the training and test data. Therefore, we expect the model to perform well on the test data even though these are unseen data.

However, this traditional vision breaks down when we do not have sufficient labeled data for the task or domain in question. Here comes the Transfer Learning to allow us to deal with these scenarios by storing the knowledge gained on the source domain and task and apply it to our problem of interest. The figure 6 illustrates the difference clearly.

4.1.2 Notation and Problem Formulation of the Domain Adaptation framework

Transfer Learning approaches can be categorized into three groups, depending on the situation of the source and target domains, and their corresponding tasks. We can talk about the inductive TL where the target task is related to the source one and the target domain contains

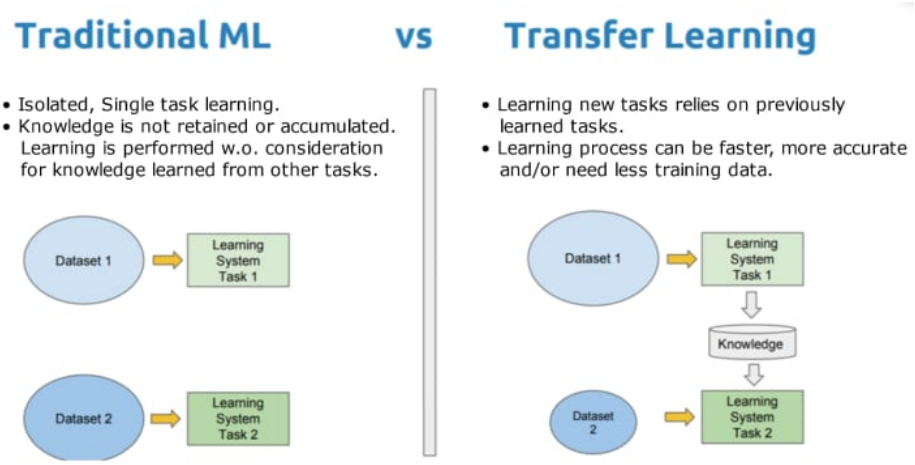

Fig. 6: Comparaison ML vs Transfer Learning [2]

some labeled target instances; the transductive TL, where labeled data is the same for the target and source domain but the tasks the model works on are different; and finally, the unsupervised TL, where both the domains and the tasks are different but somehow related, and in general, labels are not available neither for the source nor for the target domain.

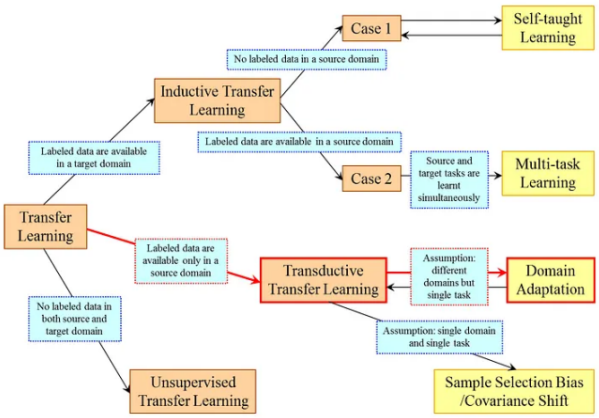

Fig. 7: Transfer Learning and fine-tuning framework [3]

About the transductive Transfer learning, his definition covers the work of Arnold et al. [4] , since the latter considered domain adaptation, where the difference lies between the marginal probability distributions of source and target data; i.e., the tasks are the same but the domains are different.

Domain adaptation is a form of transfer learning, in which the task remains the same, but there is a domain shift or a distribution change between the source and the target.

Research on how to deal with domain shift has been extensively conducted in the literature and some of them are presented later in the report. Indeed, this domain adaptation (DA) problem has received much attention. However, DA relies on a strong assumption that target data is accessible for model adaptation, which does not always hold in practice. The main

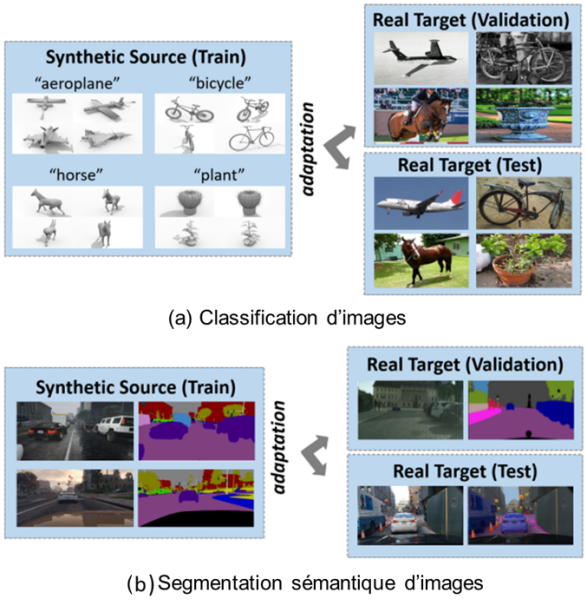

Fig. 8: Examples of data belonging to the public database VisDA2017 [5]

differences between these approaches can be seen in the table below. Indeed, the type of approach to be exploited depends above all on the conditions of the problem (e.g. mismatch in distributions or in label space) and on wether test data is accessible.

In the case of DA, the source domain is different from the target domain, and during training we have access to data belonging to both source and target domains. On the other hand, DG is more ambitious than AD. In effect, we have an offset between domains, but we don't have access to data from the target domain during model training.

Tab. 1: Comparaison between DA and DG frameworks

Approach type	Training Data	Test Data	**Problem conditions**	Access to the test data domain for the training
Domain Adaptation	$\mathcal{D}^s, \mathcal{D}^c$	$\mathcal{D}^c$	$\mathcal{D}^S \neq \mathcal{D}^c$	Yes
Domain Generalization	$\mathcal{D}^1, \cdots, \mathcal{D}^n$	$\mathcal{D}^{n+1}$	$\mathcal{D}^i \neq \mathcal{D}^j, 1 \leq i \neq j \leq n+1$	Non

Let us now formalize the problem of DA and introduce the notation :

Let $\mathcal{D}$ be the domain, which consists of a d-dimensional feature space $\mathcal{X} \subset \mathbb{R}^d$ with a marginal probability distribution $\mathbb{P}(X)$, and let $\mathcal{T}$ be the task, that consists of a label space $\mathcal{Y}$ and its conditional probability distribution $\mathbb{P}(Y \mid X)$ where $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$ are random variables. Let us assume we have two domains with their corresponding tasks such as:

- Source domain $\mathcal{D}^s = \{\mathcal{X}^s, \mathbb{P}\left(X^s\right)\}$, and its related task $\mathcal{T}^s = \{\mathcal{Y}^s, \mathbb{P}\left(Y^s \mid X^s\right)\}$

- Target domain $\mathcal{D}^t = \{\mathcal{X}^t, \mathbb{P}(X^t)\}$, and its related task $\mathcal{T}^t = \{\mathcal{Y}^t, \mathbb{P}(Y^t \mid X^t)\}$.

Note that when $\mathcal{D}^s = \mathcal{D}^t$, and $\mathcal{T}^s = \mathcal{T}^t$, we are in the traditional Machine Learning. In fact, the set of observations belonging to the source domain $\mathcal{D}^s$ will be used as the training set and the set of observations from the target domain $\mathcal{D}^t$ as the test set. However, when $\mathcal{D}^s \neq \mathcal{D}^t$ i.e. domain divergence, and/or $\mathcal{T}^s \neq \mathcal{T}^t$, we are in the Transfer Learning conditions.

In classical supervised learning approaches, the real class $y \in \mathcal{Y}$ of each training data $x \in \mathcal{X}$ is known. This information is used to learn a decision function that is able to establish the relationship between each training observation and its class. This learned function is then used to predict the classes of new observations (test data). However, if there is, for example, a mismatch in the distributions of the training and test data $P(X^{\text{train}}) \neq P(X^{\text{test}})$ and/or in the label space $y^{\text{train}} \neq y^{\text{test}}$, we find ourselves in a situation where classical learning is not sufficient.

And specifically, in Domain Adaptation we have $\mathcal{T}^s = \mathcal{T}^t$ with the presence of a domain divergence, that can be a result of :

- **A distribution shift (Homogeneous DA)** : The feature spaces are identical $\mathcal{X}^s = \mathcal{X}^t$ but the source and targets are different in terms of distributions $\mathbb{P}(X^s) \neq \mathbb{P}(X^t)$, e.g. simulated vs. real images.
- **A feature space difference (Heterogeneous DA)** : The feature spaces are different $\mathcal{X}^s \neq \mathcal{X}^t$ e.g images or the source domain consists of MRI (Magnetic Resonance Imaging) images, while the target domain contains CT (Computed Tomography) images.

In the first scenario the differences are caused by different sensors such as RGB vs. depth images, and also different types of images such as photos and paintings. In the second scenario, the differences are caused by different types of media in the source and target domains; for instance, text vs. images.

Domain adaptation methods can be categorized into several types according to factors such as the type of existing divergence between domains (seen before), the available annotations of target domain data, the type of base model used for adaptation, and the path by which domain adaptation is achieved (see figure 9).

We can also differentiate domain adaptation methods according to the annotations available in the target database:

- Supervised domain adaptation: this is the simplest case, as the labels in the target database are available. In this context, we can directly use finetuning to refine the parameters of a network initially trained with the source domain database.

- Semi-supervised domain adaptation: in this context, although most of the target data is unannotated, there are a few observations with labels. This situation is more complex than the previous one, as a large proportion of the target data is unannotated.

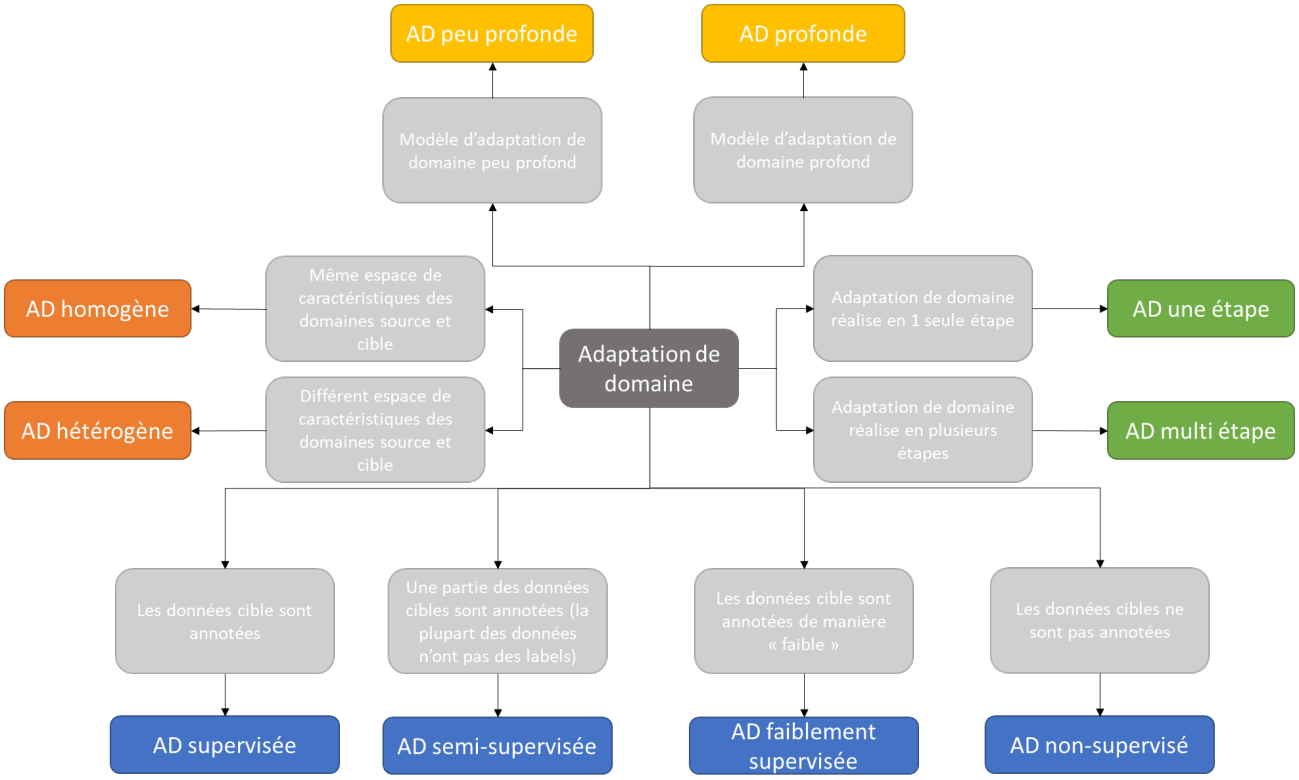

Fig. 9: Categorizing domain adaptation methods [6]

- Weakly-supervised domain adaptation: only "weak annotations" are available in the target domain. For example, in a semantic segmentation domain adaptation problem, ground truth masks are not available in the target domain, but the categories of the objects to be segmented are available.

- Unsupervised domain adaptation: this is the most studied situation in the state of the art among the recent methods, and the most complex, due to the fact that no sample of the target database is annotated. In this situation, only labels from the source database are used for model adaptation.

However, training on extremely large source datasets requires high computational resources, which is not practical, especially in real-time deployment cases. Thus, there is a high demand for source free unsupervised domain adaptation (SFUDA) methods that transfer a pre-trained source model to the unlabeled target domain without accessing any source data. In the same context, an alternative can be the N-Shot domain adaptation (zero shot, one shot).

The final form of categorization of domain adaptation techniques is based on the way in which domain adaptation is performed:

- One-step domain adaptation: if the source and target domains are directly linked, domain adaptation can be accomplished in a single step.

- Multi-step domain adaptation: if the two domains are very different, one-step domain adaptation will not be effective. Consequently, multi-step domain adaptation uses intermediate domains capable of bringing the source and target domains closer together than their initial distance.

- Between multi-step DA and continuous DA, there is gradual DA : the natural idea is to divide a large shift into multiple smaller shifts to mitigate the distribution shift issue.GDA [7] first fits a model to the source domain, then adapts it to a series of intermediate domains sequentially, and the ultimate goal is to generalize in the target domain

- Continuous domain adaptation : unlike standard domain adaptation which assumes a specific target domain, continuous domain adaptation considers the adaptation problem with continually changing target data like describe in [8].

Since the aim of this part is to analyze the most relevant methods for using real data with augmentations at most in model learning, we will focus on the domain adaptation methods most relevant to our application, i.e., homogeneous, unsupervised, deep one-step methods.

4.1.3 UDA approaches and methods : overview

In this section, without loss of generality, we first introduce terms and notations as well as a formal definition of UDA. In UDA, there are an underlying source domain distribution $p_s(x, y) \in p_{\mathcal{S}}$ and a different target domain distribution $p_t(x, y) \in p_{\mathcal{T}}$. Then, a labeled dataset $\mathcal{D}_{\mathcal{S}}$ is selected i.i.d. from $p_s(x, y)$, and an unlabeled dataset $\mathcal{D}_{\mathcal{T}}$ is selected i.i.d. from the marginal distribution $p_t(x)$.

The goal of UDA is to improve a generalization ability of a trained model in a target domain, by learning on both $\mathcal{D}_{\mathcal{S}}$ and $\mathcal{D}_{\mathcal{T}}$. We note that $\mathcal{Y} = \{1, 2, \ldots, c\}$ is the set of the class labels for discriminative tasks, e.g., classification and segmentation. In contrast, $\mathcal{Y}$ can be continuous values, sentences, images, or languages in generative tasks.

The past few years have witnessed a proliferation of UDA methods, following the rapid growth of neural network research. Popular approaches include domain alignment with statistic divergence and adversarial training, generative domain mapping, normalization statistics alignment, ensemble-based methods, and self-training, as summarized in the figure 11 :

In addition, these approaches can be combined to further enhance performance on a variety of tasks. In this section, we discuss each category in more detail as well as their combinations and connections.

4.1.4 Features level alignment

In this section, we discuss each category in more detail as well as their combinations and connections creating a domain invariant feature representation, typically in the form of a feature extractor neural network. A feature representation is domain-invariant if the features follow the same distribution regardless of whether the input data is from the source or target

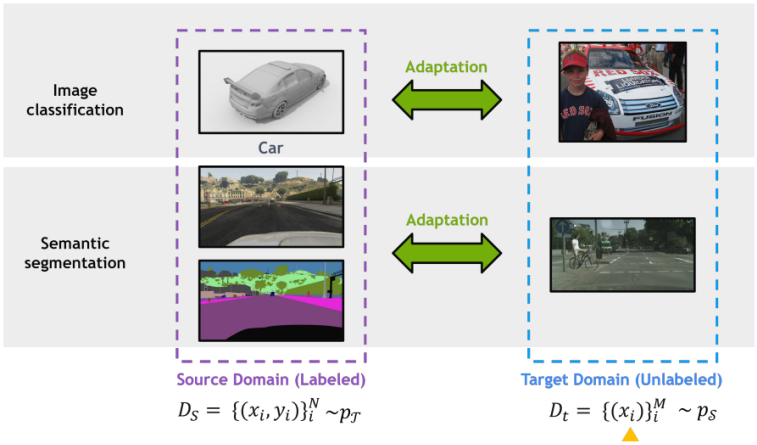

Fig. 10: Illustration of the UDA classification and segmentation with the examples on the VisDA17 challenge database. The target domain data are unlabeled, as indicated by the orange triangle. [5]

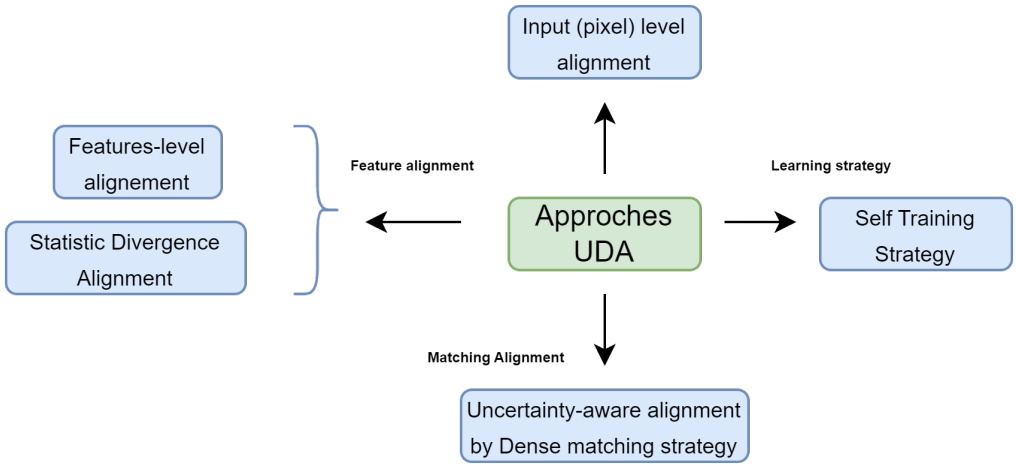

Fig. 11: Approaches of UDA

domain.

4.1.4.1 Definitions

Feature-based Adaptation techniques aim to map the source data into the target data by learning a transformation that extracts invariant feature representation across domains. They usually create a new feature representation by transforming the original features into a new feature space and then minimizing the gap between domains in the new representation space in an optimization procedure while preserving the underlying structure of the original data.

Feature-based Adaptation methods can be further categorized into the following:

- Adversarial-based methods Adaptation, which use adversarial learning to minimize the

gap between source and target domains. Generative and discriminative models are included in these approaches.

- Transformation-based Adaptation: Feature transformation transforms the original features into a new feature representation to minimize the discrepancy between the marginal and the conditional distributions while preserving the original data's underlying structure and characteristics.

- Reconstruction-based Adaptation: The feature reconstruction-based methods aim to reduce the disparity between domain distributions using a sample reconstruction in an intermediate feature representation.

4.1.4.2 Adversarial training strategy

The first approach is based on proposed adversarial learning to solve the domain adaptation problem. In most cases, the alignment component consists of a domain classifier, i.e. a classifier that indicates the original domain of the generated representations.

Unlike the others methods, adversarial DA methods follow the same approach of any adversarial method, by encouraging domain confusion through an adversarial objective with respect to a generator and a domain discriminator.

In Adversarial Discriminative Domain Adaptation (ADDA) [9], the authors review existing adversarial adaptation methods, and propose a generalized framework for adversarial DA with different choices based on the loss type, the weight sharing strategy between the two domains, and on whether they are discriminative or generative. The illustration below gives more details about this general framework.

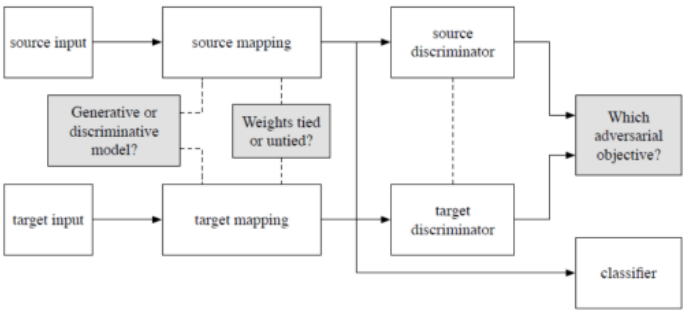

Fig. 12: Generalized framework for adversarial DA [9]

Hence, there are basically two choices that have to be taken:

- Generative or discriminative model. Firstly, the generative models are based on a Generative Adversarial Network (GAN) that is composed by two neural networks: a discriminator that tries to separate the target domain from the source domain, and a generator that tries to fool the discriminator to make the target domain look like the source one as much as possible. Secondly, the Discriminative models employ a domain confusion loss in addition

to the classification loss, that imitate the role of a discriminator in GANs, since it tries to match the distributions of he source and target domains in order to "confuse" the high-level classification layers.

- Weights tied on untied. Whether to share or not some weights between the various branch of the network

For instance, Ganin et al. proposed DANN (Domain-Adversarial Neural Network) [10], one of the first unsupervised domain adaptation methods based on adversarial learning. As can be seen in Figure 10, the proposed architecture consists of three different blocks: a generator (or extractor) of source and target image features, a domain discriminator and a classifier. The key idea of the method is to extract representations of source and target images that do not contain information about the domain of provenance. In this way, a classifier trained in a supervised way from these representations could be used for both source and target images.

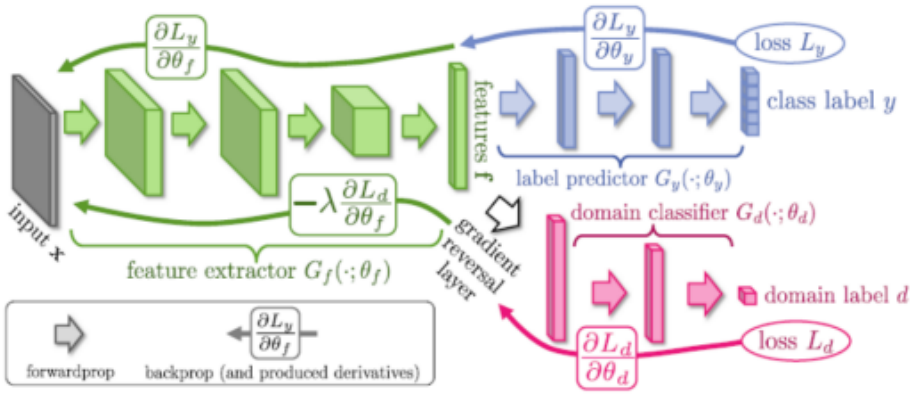

Fig. 13: Generalized framework for adversarial DA [10]

To obtain not only common representations for both domains (source and target), but also discriminative enough to succeed in the classification task, the generator (fed with source and target images) is jointly trained to fool the discriminator into extracting representations devoid of domain-related information and to minimize the classification error of the annotated source images. In addition, the discriminator is trained to correctly classify the original domain of the representations generated by the generator.

The generator and the corresponding discriminators are trained alternatively, the network propagates gradients from D to G, which would encourage G to generate similar segmentation distributions in the target domain to the source prediction.

Instead of using the same feature extractor for images from the two different domains, Tzeng et al proposed a method called Adversarial Discriminative Domain Adaptation (ADDA) [9], which has two different feature extractors (without weight sharing), one for each domain. Indeed, the proposed model is composed of a feature extractor for source data (source CNN),

a feature extractor for target data (target CNN) and a discriminator.

The network is trained in two separate phases. In the first phase, the source CNN, in association with a source classifier, is trained using annotated source images. In the second phase, the target CNN is initialized with the parameters of the pre-trained source CNN. The discriminator is trained to classify the membership domain of features extracted from each image, while the target CNN seeks to decoy the discriminator. The use of feature extractors adapted to each domain enables more appropriate representations to be obtained. Nevertheless, the authors assumed that it is possible that the source classifier may not be entirely suitable for classifying the target samples properly, due to a residual shift in the joint distributions. $(\mathbb{P}(X^s, Y^s) \neq \mathbb{P}(X^c, X^c))$.

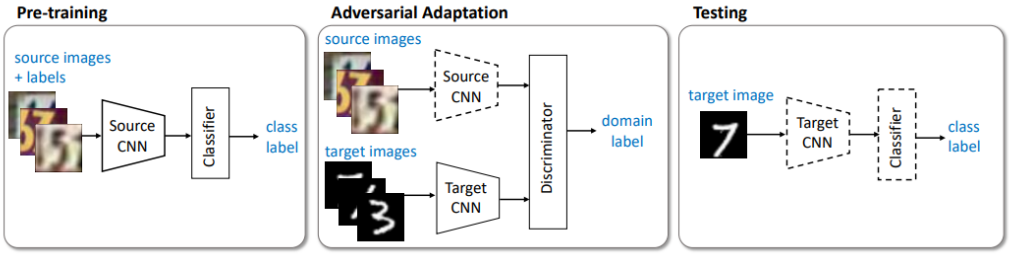

Fig. 14: Illustration of the ADDA method proposed by Tzeng et al [26].

In order to take into account the mismatch of joint distributions, the authors Tang and Jia proposed DADA [11] (Discriminative Adversarial Domain Adaptation). This method seeks to resolve the difficulty of aligning the joint distributions of features and categories between domains, which is due to the collapse mode problem induced by the use of two separate classifiers: one to classify the different categories, and the second to classify the domain (known as a domain discriminator).

Recently, the authors Chen et al. have proposed an alternative method for solving the generator collapse mode. The proposed method, named Discriminator-free Adversarial Learning Network [12] (DALN), exploits adversarial learning without explicitly using a discriminator. Unlike the DADA method, which integrates the discriminator into the category classifier classifier, DALN reuses the original category classifier as a discriminator without requiring any components, making it a very simple and effective method.

A limitation of domain-adversarial networks is that matched data distributions do not imply that the class-conditional distributions will be matched as well. Furthermore, the two loss layers produce gradients that are often in different directions. Because of this, DANNs and its alternatives can be harder to train than standard deep neural nets. Current adversarial domain adaptation models suffer from the training instability of adversarial networks.This poor performance compared with the current state of the art can be seen in the table **??**.

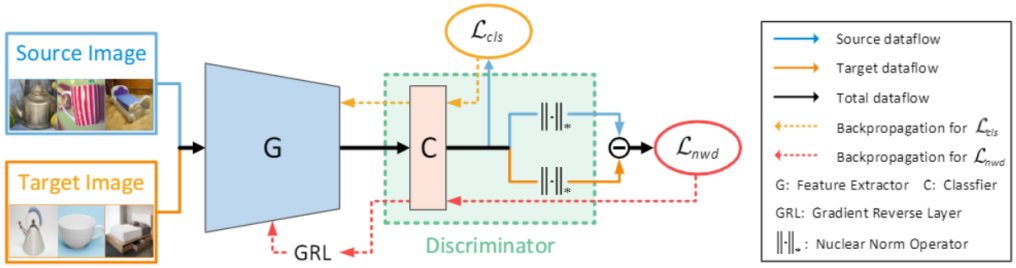

Fig. 15: Illustration of the DALN method proposed by Chen et al [12].

4.1.4.3 Statistic Divergence Alignment

Divergence-based domain adaptation methods seek to minimize the gap between the distributions of the two domains, source and target. To achieve this goal, the choice of an appropriate divergence measure is essential. Among the most widely used measures are Maximum Mean Discrepancy (MMD), Correlation Alignment (CORAL), Constrastive Domain Discrepancy (CDD) and Optimal Transport problems.

Under the assumption of a two-sample statistical test, the MMD measures the divergence of the distribution from the observed samples. Formally, the maximum mean deviation (MMD) [13] is defined as follows:

$$\mathrm{MMD}\left(\mathcal{F}, P_s, P_c\right) = \sup_{f \in \mathcal{F}} \left(\mathbb{E}_{P_s}\left[f\left(x^s\right)\right] - \mathbb{E}_{P_c}\left[f\left(x^c\right)\right]\right) \tag{1}$$

Where $\mathcal{F}$ is a space of functions $f : \mathcal{X} \to \mathbb{R}$. The MMD estimator is obtained by replacing $\mathbb{E}_{P_s}\left[f\left(x^s\right)\right]$ and $\mathbb{E}_{P_c}\left[f\left(x^c\right)\right]$ by empirical averages calculated from observations X^s and X^c :

$$\widehat{MMD}\left(\mathcal{F}, P_s, P_c\right) = \sup_{f \in \mathcal{F}} \left(\frac{1}{n_s} \sum_{i=1}^{n_s} f\left(x_i^s\right) - \frac{1}{n_c} \sum_{j=1}^{n_c} f\left(x_j^c\right) \right) \tag{2}$$

From equation 2, we can see that if the distributions of the two domains are similar, the MMD will be small. Otherwise, there will be functions f such that the difference between averages will be large. Using the function space $\mathcal{F}$, the unit ball in a Hilbert space with a kernel reproducing $\mathcal{H}$ (RKHS), i.e. $\mathcal{F} = \{f || f||_{\mathcal{H}} \leq 1\}$, the equation (1) can be expressed as the distance in $\mathcal{H}$ between the mean embeddings μ_{P_S} and μ_{P_C} :

$$\begin{aligned}
\mathrm{MMD}^2\left(\mathcal{F}, P_s, P_c\right) &= \left[\sup_{\|f\|_{\mathcal{H}} \leq 1} \left(\mathbb{E}_{P_s}\left[f\left(x^s\right)\right] - \mathbb{E}_{P_c}\left[f\left(x^c\right)\right]\right) \right]^2 \\
&= \left\|\mu_{P_s} - \mu_{P_c}\right\|_{\mathcal{H}}^2 \\
&= \mathbb{E}_{P_s}\left[K\left(x^s, x'^s\right)\right] + \mathbb{E}_{P_c}\left[K\left(x^c, x'^c\right)\right] - 2\mathbb{E}_{P_s, P_c}\left[K\left(x^s, x^c\right)\right]
\end{aligned}$$

Where $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ the core reproducing on $\mathcal{H}$.

In domain adaptation methods, the MMD can be calculated and minimized using the outputs of the different layers of the neural network. In practice, for each layer l, the biased MMD estimator is used:

$$\widehat{MMD}^2(\mathcal{F}, P_s, P_c) = \frac{1}{n_s^2}\sum_{i,j=1}^{n_s} K\left(x_i^s, x_j^s\right) + \frac{1}{n_c^2}\sum_{i,j=1}^{n_c} K\left(x_i^c, x_j^c\right) - \frac{2}{n_s n_c}\sum_{i=1}^{n_s}\sum_{j=1}^{n_c} K\left(x_i^s, x_j^c\right)$$

The methods described above are mainly based on MMD measurement to reduce the distribution gap between domains. This measure requires the choice of kernel, which is always problematic. However, other measures can also be used to measure the divergence between distributions, such as alignment by correlation (CORAL), which is calculated from the distance between the second-order statistics (covariances) of the source and target features.

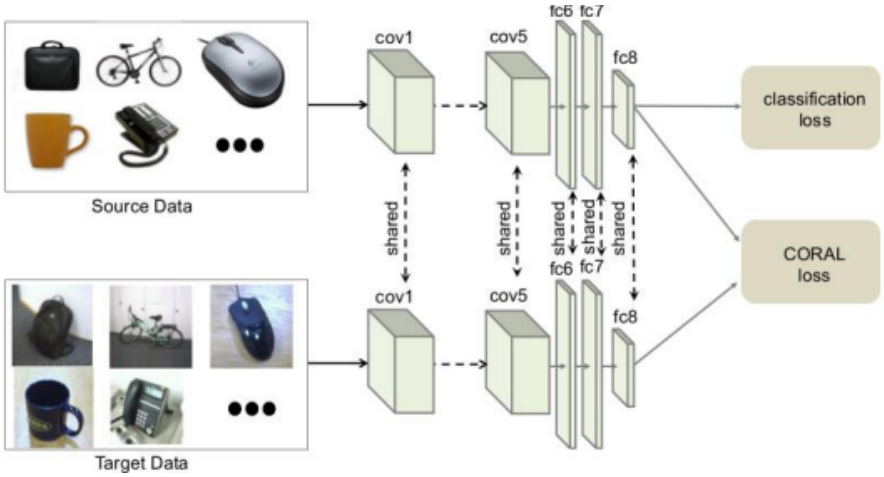

Fig. 16: Deep CORAL architecture proposed by Baochen Sun [14]

4.1.5 Self Training strategy - Notations

Adversarial training methods aim to align the distributions of source and target domain at input, feature, output, or patch level in a GAN framework. Using multiple scales or category information for the discriminator can refine the alignment. In self-training, the network is trained with pseudo-labels of images in the target domain. Most of the UDA methods pre-compute the pseudo-labels offline, train the model, and repeat the process.

The core idea is to use some pre-existing classifier Fpl (referred to as the "pseudo-labeler") to make transform predictions into pseudo-labels on a large unlabeled dataset, and then retrain a new model with the pseudo-labels. For example, in semi-supervised learning, the pseudo-labeler is obtained from training on a small labeled dataset, and is then used to predict pseudo-labels on a larger unlabeled dataset. A new classifier F is then retrained from scratch to fit the pseudo-labels, using additional regularization. In practice, F will often be more accurate than the original pseudo-labeler.

Several methods also combine adversarial and self-training, train with auxiliary tasks, or perform test-time UDA ([15], [16]).

4.1.6 Self Training strategy - DAFormer architecture

Here's an overview over the baseline UDA method for evaluating different network architectures. In UDA, a neural network g_θ is trained using source domain images $\mathcal{X}_S = \left\{x_S^{(i)}\right\}_{i=1}^{N_S}$ and one-hot labels $\mathcal{Y}_S = \left\{y_S^{(i)}\right\}_{i=1}^{N_S}$ in order to achieve a good performance on target images $\mathcal{X}_T = \left\{x_T^{(i)}\right\}_{i=1}^{N_T}$ without having access to the target labels $\mathcal{Y}_T$. Naively training the network g_θ with a categorical cross-entropy (CE) loss on the source domain

$$\mathcal{L}_S^{(i)} = -\sum_{j=1}^{H\times W}\sum_{c=1}^{C} y_S^{(i,j,c)} \log g_\theta\left(x_S^{(i)}\right)^{(j,c)} \tag{3}$$

usually results in a low performance on target images as the network does not generalize well to the target domain.

To address the domain gap, several strategies have been proposed that can be grouped into adversarial training and self-training (ST) approaches. In this work, DAFormer's authors use ST as adversarial training is known to be less stable and is currently outperformed by ST methods. To better transfer the knowledge from the source to the target domain, ST approaches use a **teacher network** h_ϕ (which we will describe later) to produce pseudo-labels for the target domain data.

$$p_T^{(i,j,c)} = \left[c = \arg\max_{c'} h_\phi\left(x_T^{(i)}\right)^{(j,c')}\right], \tag{4}$$

Additionally, a quality / confidence estimate is produced for the pseudo-labels. In the official paper, a ratio of pixels is used exceeding a threshold τ of the maximum softmax probability.

$$q_T^{(i)} = \frac{\sum_{j=1}^{H\times W}\left[\max_{c'} h_\phi\left(x_T^{(i)}\right)^{(j,c')} > \tau\right]}{H\cdot W}. \tag{5}$$

The pseudo-labels and their quality estimates are used to additionally train the network g_θ on the target domain.

$$\mathcal{L}_T^{(i)} = -\sum_{j=1}^{H\times W}\sum_{c=1}^{C} q_T^{(i)} p_T^{(i,j,c)} \log g_\theta\left(x_T^{(i)}\right)^{(j,c)} \tag{6}$$

In this particular case, the pseudo-labels are generated online due to its less complex setup with only one training stage. This is important as we compare and ablate various network architectures. In online ST, h_ϕ is updated based on g_θ during the training. Commonly, the weights h_ϕ are set as the exponentially moving average of the weights of g_θ after each training step t [17] to augmentation the stability of the predictions

$$\phi_{t+1} \leftarrow \alpha\phi_t + (1-\alpha)\theta_t. \tag{7}$$

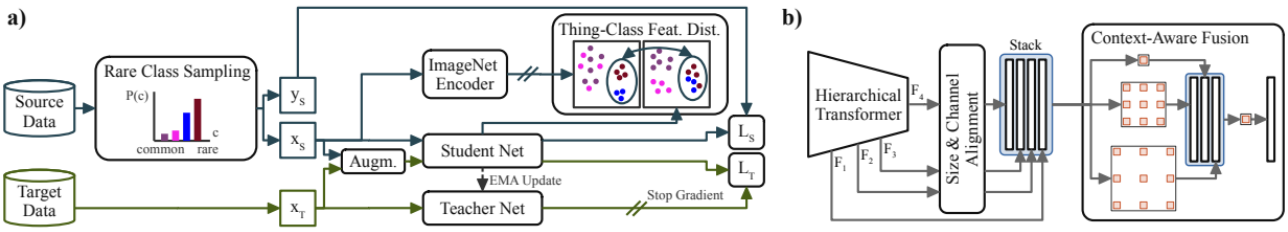

Fig. 17: Overview of our DaFormer architecture and the MIT network associated [18]

For the encoder, robustness is an important property in order to achieve good domain adaptation performance as it fosters the learning of domain-invariant features. Based on recent findings and an architecture comparison for UDA, Transformers are a good choice for UDA as they fulfill these criteria. The self-similarity operation in the self-attention mechanism provides modeling means that are potentially more adaptive and general than convolution operations.

In particular, the DaFormer architecture [18], as illustrated in the figure 17, follow the design of Mix Transformers (MiT) [19] for the teacher and Student Network, which are tailored for semantic segmentation. The image is divided into small patches of a size of 4×4 (instead of 16×16 as in ViT [20]) in order to preserve details for semantic segmentation. To cope with the high feature resolution, sequence reduction is used in the self-attention blocks. The transformer encoder is designed to produce multi-level feature maps $F_i \in \mathbb{R}^{\frac{H}{2^{i+1}} \times \frac{W}{2^{i+1}} \times C_i}$ with H the height, W the width of the image, C the number of channels and Ci the channel number of the output of stage i.

DAFormer uses the context across features from different encoder levels as the additional earlier features provide valuable low-level concepts for semantic segmentation at a high resolution, which can also provide important context information. The architecture of the DAFormer decoder is shown in the part b of the Fig 16. Before the feature fusion, it embed each F_i to the same number of channels C_e by a 1×1 convolution, bilinearly upsample the features to the size of F_1, and concatenate them. For the context-aware feature fusion, multiple parallel 3×3 depthwise separable convolutions are used with different dilation rates and a 1×1 convolution to fuse them.

One challenge of training a more capable architecture for UDA is overfitting to the source domain. To circumvent this issue, three strategies are introduced to stabilize and regularize the UDA training. The overall UDA framework is shown in the figure 17.
Firstly, UDA performance for classes that are rare in the source dataset varies significantly over different runs. Depending on the random seed of the data sampling order, these classes are learned at different iterations of the training or sometimes not at all. The later a certain

class is learned during the training, the worse is its performance at the end of the training.

To address this, Rare Class Sampling (RCS) is proposed. It samples images with rare classes from the source domain more often in order to learn them better and earlier. The frequency f_c of each class c in the source dataset can be calculated based on the number of pixels with class c

$$f_c = \frac{\sum_{i=1}^{N_S} \sum_{j=1}^{H \times W} \left[y_S^{(i,j,c)} \right]}{N_S \cdot H \cdot W} \tag{8}$$

The sampling probability $P(c)$ of a class c is defined as a function of its frequency f_c

$$P(c) = \frac{e^{(1-f_c)/T}}{\sum_{c'=1}^{C} e^{\left(1-f_{c'}\right)/T}}. \tag{9}$$

Therefore, classes with a smaller frequency will have a higher sampling probability. The temperature T controls the smoothness of the distribution. A higher T leads to a more uniform distribution, a lower T to a stronger focus on rare classes with a small fc.
When we train the network with different random seeds for the data sampling, it can achieve very different performances. Sometimes, it learns the "bicycle" class quite early during the training and then the performance is very good (figure 18). However sometimes, bad luck with the data sampling then it only learned that late in the training and the final performance is not so good.

It is not possible to rely on the randomness in order to compensate for that we basically added the rare class sampling where we sample image with rare class from the source domain more often than the ones with the common classes.This heavily augmentations the number of pixels for these rare classes during the data sampling and so improve the stability and also the performance for these rare classes.

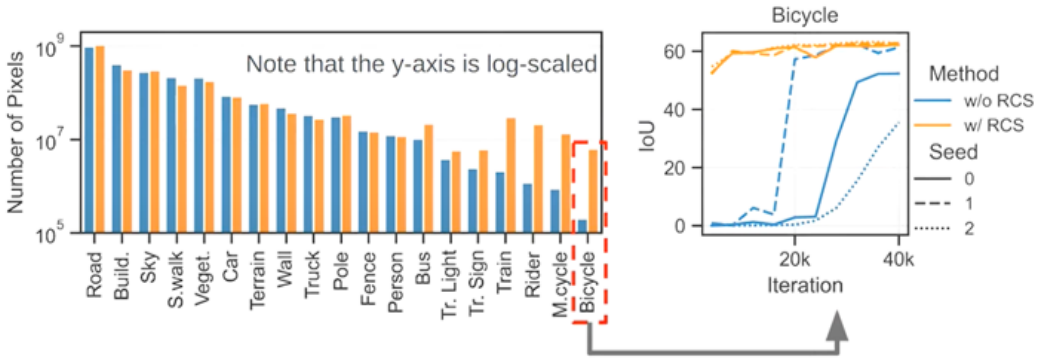

Fig. 18: Rare Class Sampling solution : Sample Images with rare classes more often [21]

Secondly, the semantic segmentation model g_θ is commonly initialized with weights from ImageNet classification to start with meaningful generic features. Given that ImageNet also

contains real-world images from some of the relevant high-level semantic classes, which UDA often struggles to distinguish such as train or bus, the ImageNet features can provide useful guidance beyond the usual pretraining.

In order to prevent this issue, we regularize the model based on the Feature Distance (FD) of the bottleneck features F_θ of the semantic segmentation UDA model g_θ and the bottleneck features F_{ImageNet} of the ImageNet model (see figure 19)

$$d^{(i,j)} = \left\| F_{\text{ImageNet}} \left(x_S^{(i)} \right)^{(j)} - F_\theta \left(x_S^{(i)} \right)^{(j)} \right\|_2 . \tag{10}$$

However, the ImageNet model is mostly trained on thing-classes (objects with a well-defined shape such as car or zebra) instead of stuff-classes (amorphous background regions such as road or sky). Therefore, the FD loss is calculated only for image regions containing thing-classes $\mathcal{C}_{\text{things}}$ described by the binary mask M_{things}

$$\mathcal{L}_{FD}^{(i)} = \frac{\sum_{j=1}^{H_F \times W_F} d^{(i,j)} \cdot M_{\text{things}}^{(i,j)}}{\sum_j M_{\text{things}}^{(i,j)}} \tag{11}$$

This mask is obtained from the downscaled label $y_{S,\text{ small}}$

$$M_{\text{things}}^{(i,j)} = \sum_{c'=1}^{C} y_{S,\text{ small}}^{i,j,c'} \cdot [c' \in \mathcal{C}_{\text{things}}] . \tag{12}$$

To downsample the label to the bottleneck feature size, average pooling with a patch size $\frac{H}{H_F} \times \frac{W}{W_F}$ is applied to each class channel and a class is kept when it exceeds the ratio r

$$y_{S,\text{ small}}^{c} = [\text{AvgPool}\,(y_S^c, H/H_F, W/W_F) > r] . \tag{13}$$

This ensures that only bottleneck feature pixels containing a dominant thing-class are considered for the feature distance.

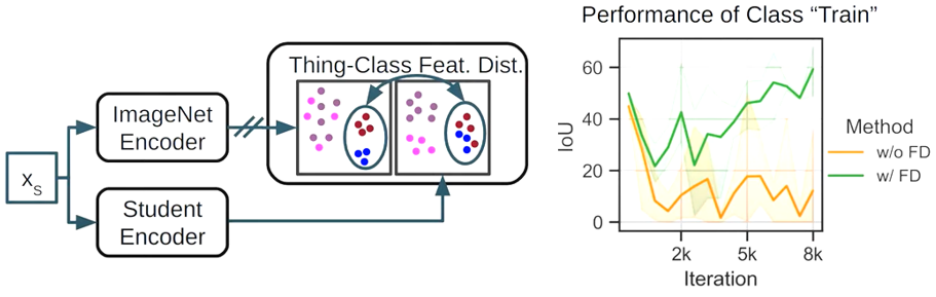

Fig. 19: Reduce distance to ImageNet features of thing-classes. Ignore stuff-classes as they are not part of the ImageNet. [21]

The overall UDA loss $\mathcal{L}$ is the weighted sum of the presented loss components $\mathcal{L} = \mathcal{L}_S + \mathcal{L}_T + \lambda_{FD}\mathcal{L}_{FD}$.

Finally, the learning Rate Warmup for UDA linearly warming up the learning rate at the beginning of the training has successfully been used to train both CNNs and Transformers as it improves network generalization by avoiding that a large adaptive learning rate variance distorts the gradient distribution at the beginning of the training. The network architecture of DAFormer consists of a Transformer encoder and a multi-level context-aware feature fusion decoder. It is enabled by three simple but crucial training strategies to stabilize the training and to avoid overfitting to the source domain.

4.1.7 Self Training strategy - DACS

Self-training has been shown to be particularly efficient if the student network g_θ is trained on augmented target data, while the teacher network h_ϕ generates the pseudo-labels using non-augmented target data for semi-supervised learning and unsupervised domain adaptation.

In this context, the DACS strategy [22] is followed with the use of color jitter, Gaussian blur, and ClassMix as data augmentations to learn more domain-robust features.

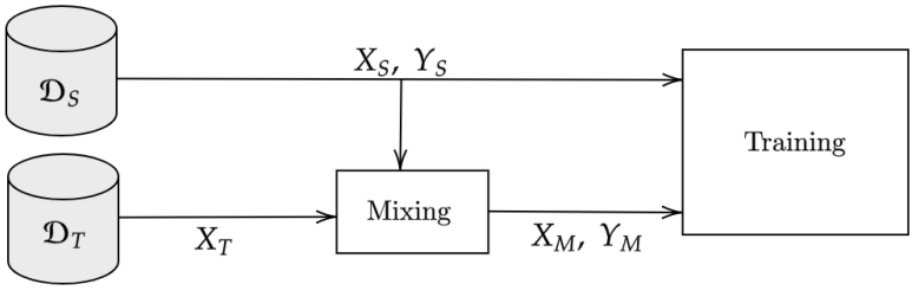

Fig. 20: Diagram showing DACS. The images Xs and Xt are mixed together, using Ys for the labels of Xs, instead of a predicted semantic map to determine the binary mask. The segmentation network is then trained on both batches of augmented images and images from the source dataset [22]

4.1.8 Source free and Domain generalization frameworks

In this subpart, the two following approaches instead targets the more general problem of source-free domain adaptation—also known as model adaptation—for semantic segmentation. In model adaptation, only (i) the model pre-trained on source images and (ii) unlabeled target images are available. This pertains to many real-world use cases, when the labeled source data is proprietary or inaccessible due to privacy concerns. The complete absence of fine ground-truth annotations represents a significant challenge, as the model can easily drift and unlearn important concepts during adaptation.

We can quote the Contrastive Model Adaptation (CMA) [23] method which leverages the reference predictions through a unified embedding space. Assuming the reference and target images are sufficiently aligned, co-located features should be similar between the two—neglecting dynamic objects and slight shifts in static content (e.g., missing leaves on a

tree). Accordingly, for a given target feature, its reference feature at the same spatial location should be closer in the embedding space than most other target features. The idea is to create such an embedding space through contrastive learning, where dense spatial embeddings of the target image serve as anchors (black patch in Fig below).

The figure 21 below shows the model adaptation architecture. The pre-trained source-model weights are used to initialize the encoder ENC and decoder DEC. Since CMA focuses on generating condition-invariant, discriminative encoder features, decoder weights are kept frozen to preserve source-domain knowledge.

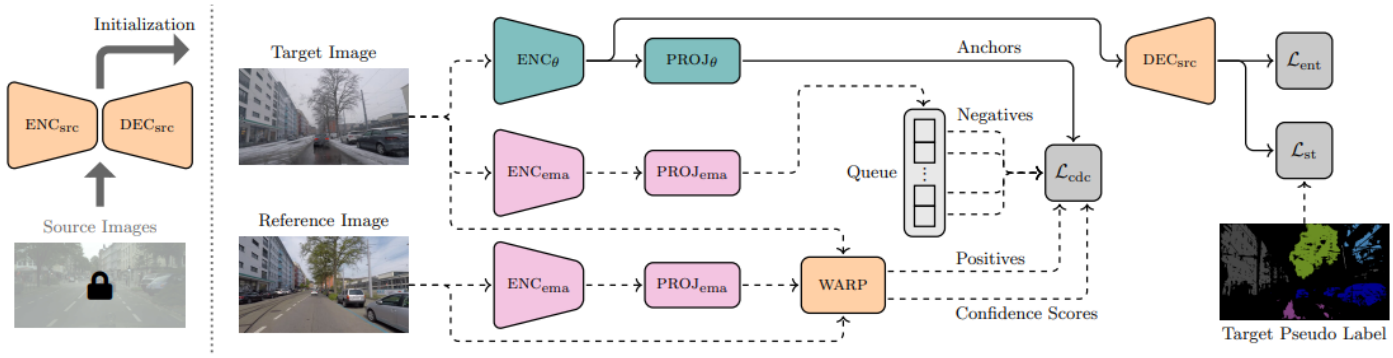

Fig. 21: Overview of the CMA architecture [23]

CMA yields state-of-the-art results for model adaptation on several normal-to-adverse semantic segmentation benchmarks. It even outperforms recent standard UDA methods on these benchmarks, despite its data handicap compared to the latter methods.

Others approaches like URMA [24] use feature corruption and entropy regularization. In deep networks, the network can learn to denoise the inputs in the initial few layers of processing. Using stronger augmentations are ill-suited for segmentation, as classification networks are expected to be invariant to such noise, whereas segmentation networks are expected to be equivariant. This can be remedied by adding structured noise to inputs such that the layout of the input objects is preserved (for example, modifying the colors of objects).

In order to avoid the annotation effort of the target domain, where the network should be deployed, the network can instead be trained on a source dataset with existing or cheaper annotations such as automatically labeled synthetic data. However, neural networks are usually sensitive to domain shifts. Therefore, domain generalization (DG) [25] methods utilize specific training strategies to improve the domain robustness so that the network better generalizes from the source to unseen target domains. Unlike unsupervised domain adaptation (UDA), the network is additionally adapted to the known target domain using unlabeled target images.
To improve the domain generalization of the model, typical approaches include removing style information from the features and diversifying the source domain training data with addi-

tional styles. These approaches opt for the latter group as it enables benchmarking different network architectures without modifying the underlying architectures. Exist SHADE [26] which diversify the style of encoder features via AdaIN [27] by interpolating new styles from a set of basis styles. Moreover, while DAFormer was originally designed for UDA, it was extended both to DG [28]. The proposed components display significant improvements in DG, demonstrating their effectiveness beyond UDA.

4.1.9 SOTA comments

The majority of the approaches seen previously rely on discrepancy minimization, adversarial training, or self-training. The first group minimizes the discrepancy between domains using a statistical distance function such as maximum mean discrepancy, correlation or Fourier alignment, or entropy minimization. In self-training, pseudo-labels are generated for the target domain based on predictions obtained using confidence thresholds or pseudolabel. prototypes.

To increase the robustness of the self-training, consistency regularization is often applied to ensure consistency over different data augmentations, domain mixup (DACS), different crops (MIC - Masked Image Consistency for Context-Enhanced Domain Adaptation [29]), the addition of a reference image and a refinement module (Refign - Align and Refine for Adaptation of Semantic Segmentation to Adverse Conditions [30]) or differents resolutions improvements (HRDA - Context-Aware High-Resolution Domain-Adaptive Semantic Segmentation [31]). The self training strategies based on DAFormer approach achieve significant performance improvements in all of the UDA tasks and constitute the state of the art nowadays.

4.2 Choice of Methods and technical environnement

In order to better understand the framework of the internship assignment, in the following chapter we will discuss the technical framework, the tools used during the assignment and describe the internship strategy followed.

4.2.1 Semantic segmentation task in deep learning framework and metrics

The internship topic concerns the development of unsupervised domain adaptation methods for semantic segmentation algorithms.

Image segmentation is an essential component in many visual understanding systems. It involves partitioning images (or video frames) into multiple segments or objects. Segmentation plays a central role in a broad range of applications, including therefore autonomous vehicles (e.g., navigable surface and pedestrian detection). Over the past few years, deep learning (DL) models have yielded a new generation of image segmentation models with remarkable performance improvements —often achieving the highest accuracy rates on popular benchmarks— resulting in a paradigm shift in the field.

Image segmentation can be formulated as a classification problem of pixels with semantic labels (semantic segmentation) or partitioning of individual objects (instance segmentation). Semantic segmentation performs pixel-level labeling with a set of object categories (e.g., human, car, tree, sky) for all image pixels, thus it is generally a harder undertaking than image classification, which predicts a single label for the entire image. Instance segmentation extends semantic segmentation scope further by detecting and delineating each object of interest in the image (e.g., partitioning of individual persons).

For example, figure 22 presents semantic predictions of a popular deep learning model, DeepLabv3 on an image of the Cityscapes DataSet.

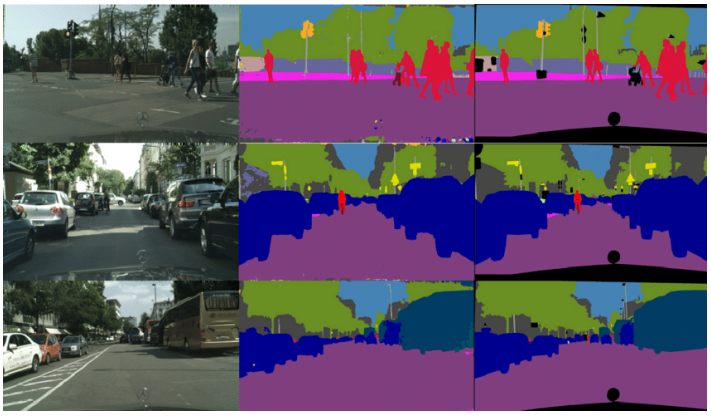

Fig. 22: Example of segmentation in the test set of cityscapes. first column: input RGB images; second column: prediction of our experiments; third column: ground-truth

In the semantic segmentation framework, exist a particular parameter type of "reduce-zero-label" a Boolean which is default to False. It is used to ignore the dataset label 0. The specific method is to change label 0 to 255, and subtract 1 from the corresponding number of all the remaining labels. At the same time, set 255 as ignore index in the decode head, which means that it will not participate in the loss calculation. If we are only two classes, you should not use reduce-zero-label which is reduce-zero-label=False.

In our case, all the dataset used are composed of several segmentation classes. We want that reduce-zero-label equals to True in order to have a a label 0 which represent a class named "void" representing the set of pixels that do not belong to any class defined beforehand.

Ideally, a model should be evaluated in multiple respects, such as quantitative accuracy, speed (inference time), and storage requirements (memory footprint). However, most of the research works so far, focus on the metrics for evaluating the model accuracy. Below we summarize the metrics used in this internship for assessing the accuracy of segmentation algorithms. Although quantitative metrics are used to compare different models on benchmarks, the visual quality of model outputs is also important in deciding which model is best.
Pixel accuracy simply finds the ratio of pixels properly classified, divided by the total number of pixels. For $K + 1$ classes (K foreground classes and the background) pixel accuracy is

defined as equation 14:

$$\text{PA} = \frac{\sum_{i=0}^{K} p_{ii}}{\sum_{i=0}^{K} \sum_{j=0}^{K} p_{ij}}, \tag{14}$$

where p_{ij} is the number of pixels of class i predicted as belonging to class j.

Mean Pixel Accuracy (MPA) is the extended version of PA, in which the ratio of correct pixels is computed in a per-class manner and then averaged over the total number of classes, as in equation 15:

$$\text{MPA} = \frac{1}{K+1} \sum_{i=0}^{K} \frac{p_{ii}}{\sum_{j=0}^{K} p_{ij}}. \tag{15}$$

Intersection over Union (IoU) or the Jaccard Index is one of the most commonly used metrics in semantic segmentation. It is defined as the area of intersection between the predicted segmentation map and the ground truth, divided by the area of union between the predicted segmentation map and the ground truth:

$$\text{IoU} = J(A, B) = \frac{|A \cap B|}{|A \cup B|}, \tag{16}$$

where A and B denote the ground truth and the predicted segmentation maps, respectively. It ranges between 0 and 1 .

Mean-IoU is another popular metric, which is defined as the average IoU over all classes. It is widely used in reporting the performance of modern segmentation algorithm and used as a reference metric in our study. Mean-IoU metrics does not replace IoU per class metrics but its is complementary.

4.2.2 MMSEG, a pytorch framework in a distributed environment

We run all of our experiments based on mmsegmentation framework of the MMLab toolbox. You can access the mmsegmentation documentation via this link [32] and discover all the possibilities offered by this framework. The framework mmesegmentation is decomposed into different components and one can construct a customized semantic segmentation pipeline by combining different modules. It supports training on multiple GPUs and multiple open source Datasets like Cityscapes, VOC12aug etc. It has almost every State of the art model pre-configured. In the context of the internship, all experiments were carried out using public github repositories whose architecture and script format is based on MMsegmentation.

We usually define a neural network in a deep learning task as a model, and this model is the core of our semantic segmentation solution. MMSegmentation abstracts a unified model BaseModel to standardize the interfaces for training, testing and other processes. All models implemented by MMSegmentation inherit from BaseModel, and in MMSegmentation anf forward pipeline and some functions for the semantic segmentation algorithm were addeed. As illustrated in the figure 23, MMSegmentation abstract the network architecture as a Segmentor, it is a model that contains all components of a network. MMSegmentation wraps

BaseModel and implements the BaseSegmentor class, which mainly provides the interfaces forward, train step, val step and test step.

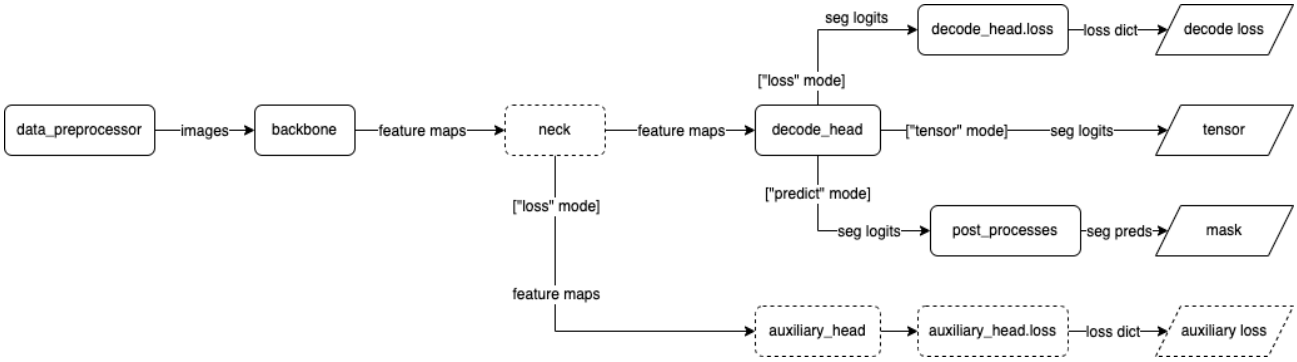

Fig. 23: The forward of BaseSegmentor class returns losses or predictions of training, validation, testing, and a simple inference process.

From a data processing point of view, DataLoader is an essential component in training and testing pipelines of MMSegmentation. Conceptually, it is derived from and consistent with PyTorch. DataLoader loads data from filesystem and the original data passes through data preparation pipeline, then it would be sent to Data Preprocessor.

In the context of the internship and taking advantage of Safran Tech's gpus, we implement distributed training which uses MMDistributedDataParallel module of MMsegmentation. All outputs (log files and checkpoints) will be saved to the working directory, which is specified by workdir in the config file. After training a model on a training set, we test by default the model on the validation set after some iterations. To avoid overfitting on the training dataset, we use Early stopping, a form of regularization which keeps track of the validation loss, if the loss stops decreasing for several epochs in a row the training stops. Example of the pair train/validation loss obtained for the DAFormer task is represented in the figure 24.

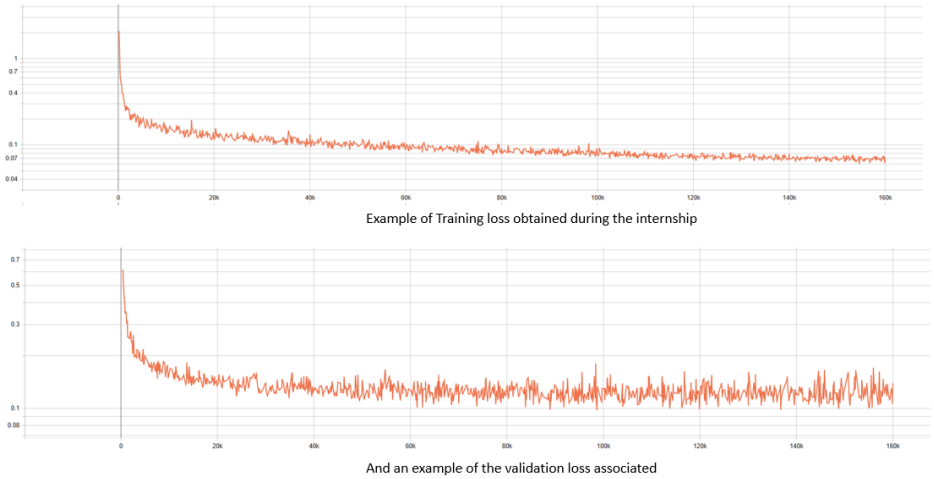

Fig. 24: Example of Train/Validation losses obtained during the internship

The default learning rate in config files is for 4 GPUs and 2 img/gpu (batch size = 4x2 = 8). Originally, it is so memory challenging to fit 2 img/gpu for semantic segmentation. Howver, in our particular case, large images (e.g. 1024x512) are fitted as opposed to cifar or even imagenet case. Equivalently, you may also use 8 GPUs and 1 imgs/gpu since all models using cross-GPU SyncBN.

SyncBC means Synchronized Multi-GPU Batch Normalization. Firstly, BN layer was introduced in this paper [33], which dramatically speed up the training process of the network (enables larger learning rate) and makes the network less sensitive to the weight initialization. However, standard implementations of BN in public frameworks (such as PyTorch) are unsynchronized, which means that the data are normalized within each GPU like represented in the figure 25. Therefore the working batch-size of the BN layer is BatchSize/nGPU (batch-size in each GPU).

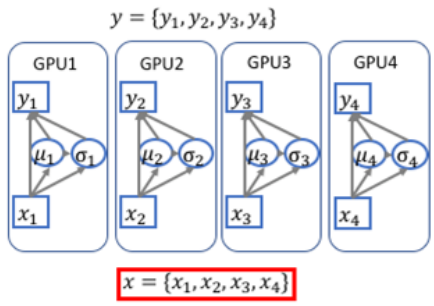

Fig. 25: Batch normalization in training mode

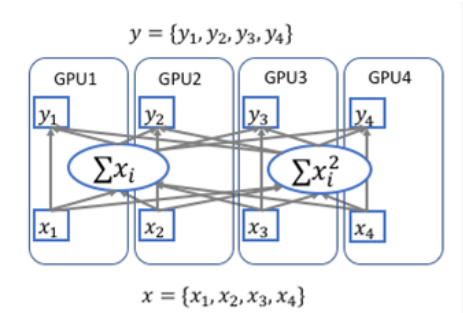

Fig. 26: Synchronized Batch normalization in training mode

Since the working batch-size is typically large enough for standard vision tasks, such as classification and detection, there is no need to synchronize BN layer during the training. The synchronization will slow down the training.

However, for the Semantic Segmentation task, the state-of-the-art approaches typically adopt dilated convoluton or self attention, which is very memory consuming. The working bath-size can be too small for BN layers (2 or 4 in each GPU) when using larger/deeper pre-trained networks.

To implement synchronized cross-gpu batch normalization (SyncBN) on PyTorch, MMSegmentation use NVIDIA NCCL Toolkit. The implementation only requires synchronizing one time by applying a simple strategy: for the N number of given input samples $X = \{x_1, \dots x_N\}$, the variance can be represented by

$$\begin{aligned} \sigma^2 &= \frac{\sum_{i=1}^{N} (x_i - \mu)^2}{N} \\ &= \frac{\sum_{i=1}^{N} x_i^2}{N} - \frac{\left(\sum_{i=1}^{N} x_i\right)^2}{N^2}, \end{aligned} \tag{17}$$

where $\mu = \frac{\sum_{i=1}^{N} x_i}{N}$. We first calculate $\sum x_i$ and $\sum x_i^2$ individually on each device, then the global sums are calculated by applying all reduce operation over multiple gpus with synchronization. The global mean and variance are calculated using equation 17 and the normalization is performed for each sample $y_i = \gamma \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta$. Similarly, we synchronize once for the gradients of $\sum x_i$ and $\sum x_i^2$ during the back-propagation. Also by default, during training this layer keeps running estimates of its computed mean and variance, which are then used for normalization during evaluation. The running estimates are kept with a default momentum of 0.1. The training mode with SyncBatchNorm is illustrated in the figure 26.

4.2.3 Definition of the research strategy

Discovering a new Pytorch code environment and familiarize yourself with the UDA framework meant that I had to address the problem set out in the paragraph in a methodical and experimental way.
The first step was to familiarize yourself with the mmseg framework by understanding and testing the segformer architecture and the UDA framework.
SegFormer [34] is a simple, efficient yet powerful semantic segmentation framework which unifies Transformers with lightweight multilayer perceptron (MLP) decoders. It is based on Mix Transformer encoders (MiT) and it is partly inspired by Vision Transformer (ViT) but tailored and optimized for semantic segmentation. The idea was to implement the paper and test it on various public data sets such as Cityscapes [35] or ACDC [36].

Then, the UDA architecture was studied and implemented. The majority of the approaches seen in the state of the art are based on discrepancy minimization, adversarial training, or self-training. The self training strategies based on DAFormer approach achieve significant performance improvements in all of the UDA tasks and constitute the state of the art nowadays. They are characterized by the DaFormer base and additionnal plug-ins in order to reinforce the robustness of the pseudo-labels. In this context, the DAFormer strategy was chosen to apply the UDA on different data. The more recent MIC and HRDA approaches were not tested, for reasons of cost (HRDA requires high-resolution images with very high computing power) or time (MIC). The idea was to implement the paper of DAFormer and test it on various use cases : cityscapes to ACDC, cityscapes to RUGD to name but a few. The challenge is to know whether the results are consistent with those found in the research paper and interesting to note. For example, the results of domain adaptation from Cityscapes to ACDC can be put into perspective with those of Cityscapes to RUGD. Similarly, domain adaptation results from Cityscapes to RUGD can be compared with those from RUGD to Samba to highlight the domain shift between datasets. This first experimentation part is shown in yellow in the figure 28 below .

Secondly, after this first phase, more in-depth work was carried out on two levels:

- both on the domain adaptation between RUGD and Samba data and to clarify the relevance of choosing RUGD as a proxy for subsequent experiments on Samba data.

- as well as the data augmentation approach and the evaluation of domain adaptation between the RUGD data set and "RUGD augmented".

All these results have been compared with those obtained in the previous phase to define whether or not RUGD dataset can be representative of Samba data, and whether experimenting with data augmentation on RUGD data can be a good solution for simulating cases of domain adaptation between Safran datasets (including Samba). This second experimentation part is shown in pale brown in the figure 28.
Three types of segmentation experiments have been launched for the preliminaries steps and for the research improvements :

- Oracle results: directly training SegFormer using training target data
- Source only results: Train a SegFormer on a given data set then infer on another dataset
- DaFormer between two domains represented by two datasets

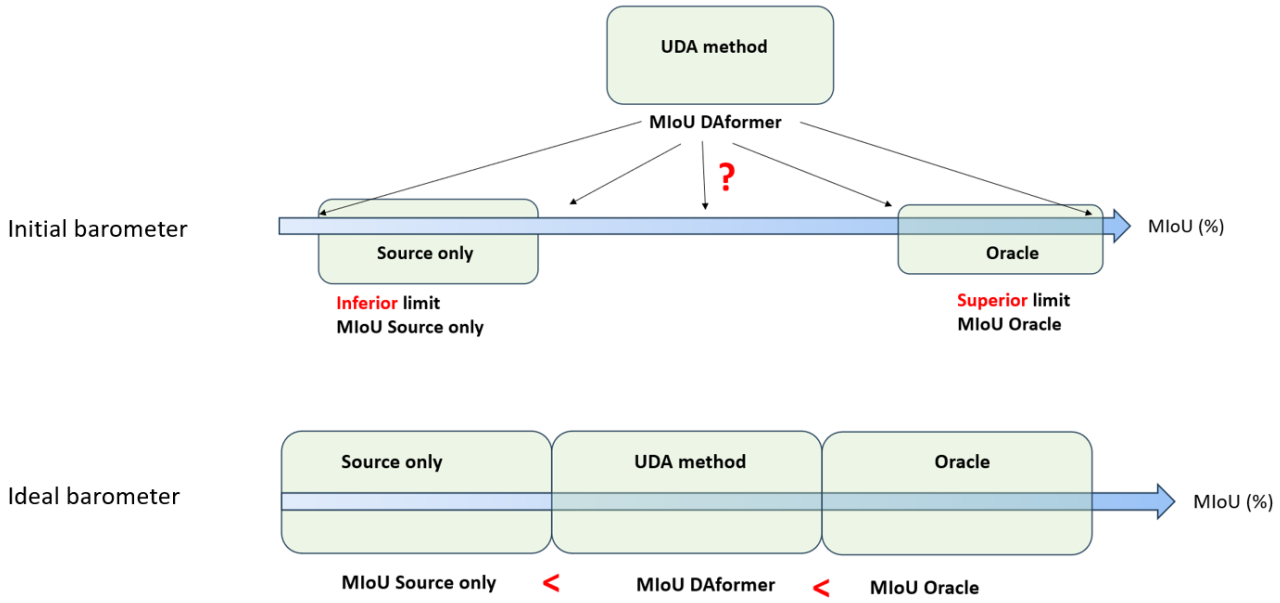

Fig. 27: Diagram of the three types of experimentations represented as a barometer

These three types of experimentation constitute our barometer for evaluating our DAFormer model applied to several cases. This barometer can be illustrated as follows, including the evaluation metrics chosen. Ideally, the MioU of the DAFormer should be higher than that of the source only but lower than that of the oracle to ensure consistency.

In addition to interpreting the mIoU metric, Lukas Hoyer, one of the authors of the DAFormer, has defined a new indicator called "relative UDA performance" in the paper [18], which is the ratio of performance between the UDA performance and the supervised oracle :

$$\frac{MIoUUDA}{MIoUOracle}$$

That's basically a measurement like how well is the network architecture actually doing its capacities. The more maximal the better, meaning that we significantly narrowed the gap between unsupervised domain adaptation and supervised learning.

However, this metric does not take into account the performance that the source-only model would have achieved.

In the context of this internship, the idea was to propose a global metric that we can call "UDA gain" as a reference to evaluate the UDA gain . To do this, the previous metric can be reuse, but normalized in the denominator and numerator by the score of the free source experiment :

$$\frac{MIoUUDA - MIoUSourceonly}{MIoUOracle - -MIoUSourceonly}$$

The denominator refers to the performance gap between a model without access to target data and a model with access to target data. The numerator indicates the extent to which the UDA has closed this gap. If you get a UDA gain strictly inferior to zero, the UDA doesn't add any value and doesn't perform as it should.

These tests will require intermediate pre-processing steps, including label transformations according to the data used, and class redefinitions to ensure domain adaptation.
All these steps can be summarized in the figure 28, which perfectly sums up all the tasks carried out during the internship :

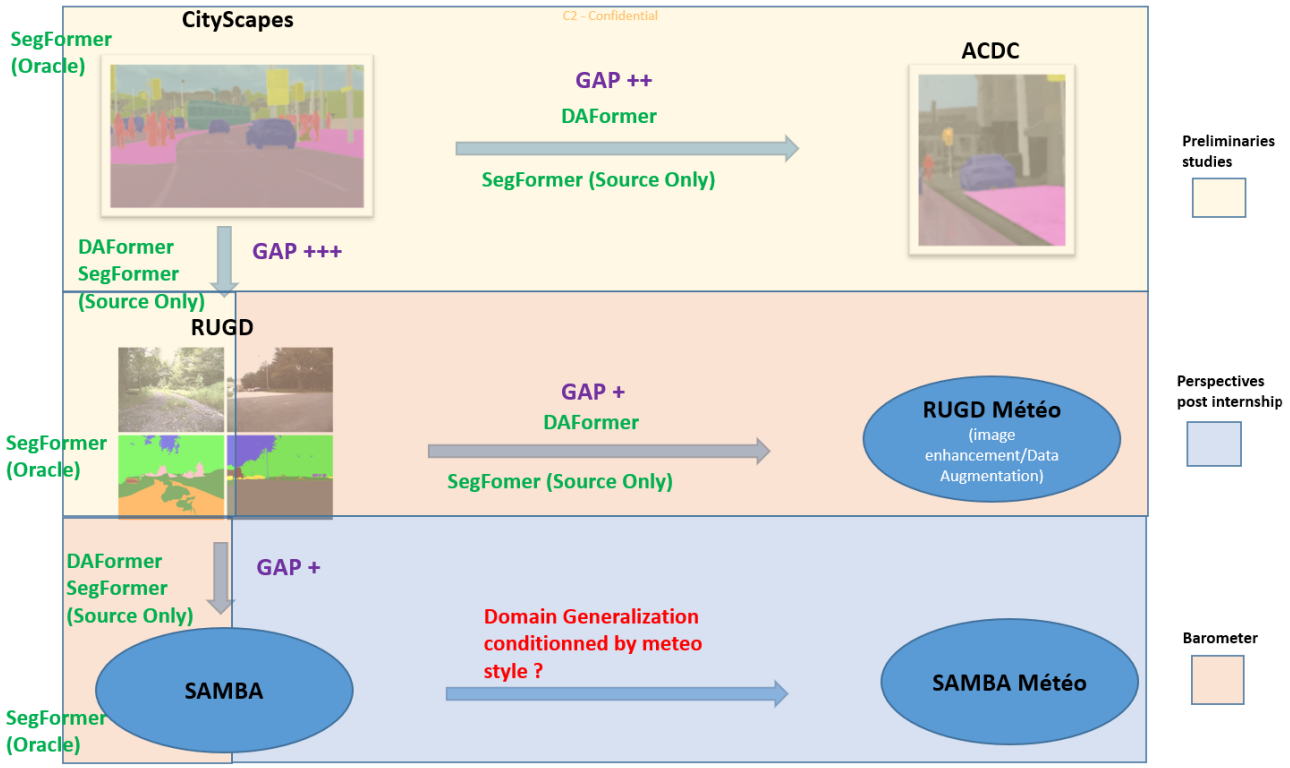

Fig. 28: Adopted strategies and experimentations during the internship

The blue part of the diagram represents the different questions and perspectives of the internship, and will be explained later in the report.

5 Familiarization and initial results

This chapter presents the first results obtained in the context of our problem to be solved. Class mapping and image pre-processing techniques will be presented, along with their influence on the results obtained.

5.1 SegFormer experimentations

As explained in the paper [34], the proposed MLP decoder of Segformer architecture aggregates information from different layers, and thus combining both local attention and global attention to render powerful representations. We show that this simple and lightweight design is the key to efficient segmentation on Transformers.

To facilitate efficient discussion, a code name was assigned from B0 to B5 for MiT encoder, where B0 is the smallest model designed for real-time, while B5 is the largest model designed for high performance. The appendix A.2 shows the detailed information of our MiT series. For these experimentations, the MiT B0 was choosen knowing that the idea is above all to understand the architecture of the model and reproduce results that are consistent with those defined in the state of the art.

As explained in the appendix A.3, Cityscapes is decomposed of 24 classes and assessing the performance of vision algorithms for major tasks of semantic urban scene understanding in nominal conditions.

On the contrary, RUGD dataset focuses on semantic understanding of unstructured outdoor environments for applications in off-road autonomous navigation. Many of the classes of RUGD are sparsely represented in the ground truth annotation (as noted by the need for an inset to better visualize some of the classes). This is in part due to the complete absence of labels in many videos for labels such as rock bed, bicycle, bridge and picnic table. RUGD data is divided into environment types and the distribution of data for each environment into selected train, test, validation sets. All this information is described in the research paper [37] and sum up in the appendixe A.4. I strictly followed the strategy of splitting, validating, testing the environments of the RUGD research paper. For example, "trail" type environments which are offroad type environments were placed in the split validation to study the model's ability to generalize well. Thereafter, it will be interesting to test different distributions of environments in the different splits.

Méthode	MIou on train set (%)	MIou on test set (%)
SegFormer on Cityscapes	79.93	78.15
SegFormer on RUGD	46.43	37.1

Tab. 2: Global training/validation results of the Segformer model on Cityscapes and RUGD

The results of tab 2 obtained prove the segmentation efficiency of the Segformer model on the Cityscapes dataset, with a MIoU metric reaching 78% on the test data. Conversely, the 37% MIoU score on the RUGD test data should be seen in the context of the sparse nature of the RUGD labels for certain classes.
Indeed, if we look in more detail at the metrics per class, we notice the "nan" or "0%" of metrics for classes absent in both RUGD test environments. According to the appendix A.5, the "dirt", "asphalt", "personn" and "concrete" classes are very rarely present and labeled in the "park" and "trail-5" test environments. These environments are wilderness areas, tall grass forests with occasional dirt tracks (trail -5) or parking spaces in the middle of forest areas. Automatically, without ground truth as a reference, the model cannot perform on these classes and returns very low metric scores or nan values.

The qualitative results with the predicted segmentation masks can be found in the appendix A.6 for the Cityscapes and in the appendix A.7 for the RUGD DataSet.

Thereafter, it will be interesting to test different distributions of environments in the different splits.

5.2 Pre-processing mapping des labels/classes and tests

Before testing the DAFormer on the adaptation between Cityscapes and RUGD, as well as source-only tests, it is necessary to reflect on the preprocessing steps to be carried out, which are essential before embarking on experimentation. The main steps are described below:

- customize datasets by reorganizing data : A training pair is built and consist of the files with same suffix in imgdir then anndir folders. Some datasets don't release the test set or don't release the ground truth of the test set, and we cannot evaluate models locally without the ground truth of the test set, so we set the validation set as the default test set in config files. The annotations are images of shape (H, W), the value pixel should fall in range [0, numclasses - 1]. We use 'P' mode of pillow to create our annotation image with color.

- Labels transformation: the initial labels of each dataset are transformed into "train-ids", each corresponding to a class number. This label transformation, launched offline, enables the labels of the various datasets to correspond to each other.

- Class mapping: As with labels, source-only and daformer experiments require a mapping of classes between source and target. The idea is therefore to create common meta classes from existing RUGD and Cityscapes classes. The choice of meta classes was made on the basis of common classes between RUGD and Cityscapes (e.g. building, pole, person, fence,

sign) and by grouping certain classes into a single one (tee and bush composed the meta class Vegetation). This is illustrated in the table below with the RUGD class mapping. But it's exactly the same mapping of Cityscapes classes. These meta classes were defined progressively after several tests with Segformer and source-only.

The autonomous vehicle experts on the CasPer team also helped me to make these choices.

Méta classes	**Void**	**Flat**	**vehicle**	**Vegetation**	**Terrain**	**Pole**	**Sign**	**Sky**	**Construction**
Initial classes	Void Water Container Log person Rock Pic nic table	Asphalt	Vehicle Bicycle	Tree Bush	Dirt Sand Grass Gravel Mulch Rock-bed Concrete	Pole	Sign	Sky	Building Fence

Fig. 29: Class mapping overview table for Rugs classes

The relevance of the class mapping was then evaluated to define whether :

- the Segformer's oracle performance on the meta classes was as good as that on the initial classes
- the Segformer trained on Cityscapes with the initial classes is able to generalize on Cityscapes but with the meta classes.
- Same forRUGD

All the results obtained are summarized in the table 3 below:

Méthode	MIou on train set (%)	MIou on validation set (%)
Source Only CS meta class	78.43	77.35
Source Only RUGD meta class	64.74	61.26

Tab. 3: Global training/validation results of the Segformer model on Cityscapes and RUGD

An example of a segmentation mask predicted for meta classes on the RUGD dataset is shown in the appendix A.8. The various results obtained show that, despite the loss of semantic information in remapping, the evaluation metrics of a segformer model applied as a source-only model on meta classes remain equivalent to those of a source-only model on initial classes. However, this observation does not apply to the RUGD dataset. As RUGD classes are initially very sparse, being able to merge certain classes together and create new classes augmentations performance. Classes with initially little labeled data are merged to form a meta-class containing more labeled data. The boundary between meta-classes is thus better defined, and MIoU scores are thus augmentationd.

5.3 DAFormer experimentations preliminaries

Thanks to the previous mapping of classes to RUGD, it is now possible to test the DAFormer on Cityscapes to RUGD. Besides, It is interesting to compare this test with the DAFormer experiment from Cityscapes to ACDC, and to identify several interpretations.

Méthode	MIoU on train set (%)	MIoU on test set (%)
Cityscapes to RUGD		
Source only RUGD	19.12	18.35
DAFormer CS to RUGD	25.03	22.09
Oracle RUGD	32.89	30.41
Cityscapes to ACDC		
Source only ACDC	47.29	43.81
DAFormer CS to ACDC	57.93	53.59
Oracle ACDC	79.74	76.26

Tab. 4: Barometer results of the DAFormer model for two cases : CS to RUGD and CS to ACDC

For both experiments, the oracle results remain superior to the UDA results. And the latter exceed those of source-only. So we're in the barometer's ideal case. We reach a "UDA gain" of

$$\frac{22.09 - 18.35}{30.41 - 18.35}$$

equals to 33 % for the daformer to RUGD and a "UDA gain" of

$$\frac{53.59 - 43.81}{76.26 - 43.81}$$

equals to 30 % for the daformer to ACDC. For the two cases, this shows that the UDA, through the DAFormer, makes it possible to gain in performance on the semantic segmentation task compared to source-only.
MIoU's results for DAFormer are, however, lower for the CS to RUGD gap. This can be explained by the higher gap from cityscapes to RUGD, given that the RUGD dataset is made up of unstructured outdoor environments for applications in off-road autonomous navigation. ACDC, on the other hand, targets semantic understanding of driving scenes in adverse visual conditions. Recordings performed in Switzerland, primarily in urban areas but also on highways and in rural regions to a lesser extent which most closely matches the semantic content of the Cityscapes dataset.

From the class point of view, all results are shown in the appendix A.9. The results of the daformer's class metrics are highly unequal, with :

- a fairly high result for the vegetation class (75.67%)

- very low results for the "sign" and "void" classes (less than 5%). These results can be explained, as previously mentioned, by the low number of data labelled "sign" and "void" in the RUGD environments chosen as test data. This is despite class mapping.

An example of the segmentation mask predicted by the DAFormer on the Cityscapes dataset and on the RUGD data set are shown in the appendix A.10 and in the appendix A.11

All these preliminary results already highlight the characteristics of the RUGD dataset and the potential difficulties of generalizing the DAFormer model to this data due to the domain gap and specific semantic characteristics. These results will be taken up in the next chapter in order to address the problematic of the internship.

6 Research improvements and answers of the problem

The initial aim of the internship is to evaluate the relevance of the unsupervised domain adaptation framework to the problem of adapting domains to harsh weather conditions. The challenge is to be able to evaluate this framework on RUGD data as a proxy for Samba as defined at the beginning of the report.
On the other hand, RUGD is a dataset made up of images captured in off-road but nominal environments. The "RUGD weather" domain (RUGD data captured in adverse weather conditions) must exist but we do not have access to it.
In order to carry out our initial UDA task on adverse data, there is therefore a strong need to artificially augment the RUGD data using weather-specific data augmentation techniques.

6.1 State of the art summary of the data augmentation for adverse conditions

In an autonomous driving system, perception is crucial, the weather can be diverse and can change abruptly, causing significant degradation in perception, resulting in ineffective manoeuvres. In order to improve vision tasks in adverse weather, deep-learning-based models typically require extensive datasets captured in such conditions - the collection of which is a tedious, laborious, and costly process. However, recent developments allow the synthesis of highly realistic scenes in multiple weather conditions. To this end, we can introduce a few approaches of using synthesised adverse condition datasets in autonomous driving.

For image translation framework, existing neural style transfer methods require reference style images to transfer texture information of style images to content images. However, in many practical situations, users may not have reference style images but still be interested in transferring styles by just imagining them. In order to deal with such applications, a new

framework enables a style transfer 'without' a style image, but only with a text description of the desired style.

Using pre-trained text-image embedding models, these method usually deliver semantic information of text condition to the visual domain. However, these methods often have disadvantages in that semantics are not properly reflected due to the performance limitations of the embedding model, and the manipulation is restricted to a specific content domain (such as human face) as the method heavily rely on pre-trained generative models. To address this, a novel image style transfer method deliver the semantic textures of text conditions using recently proposed text-image embedding model of CLIP, Clip Styler [38].

First, instead of optimizing the loss by using the image directly, a patch-wise CLIP loss is used to guide the network to function as a brush-stroke.By applying this patch-wise CLIP loss, it is possible to transfer styles to each local area of the content image. Furthermore, the augmentation induces the patch style to be more vivid and diverse.

On the other hand, large text-to-image models achieved a remarkable leap in the evolution of AI, enabling high-quality and diverse synthesis of images from a given text prompt. ClipStyler produces images that exhibit characteristic styles of the input text prompt. However the stylized images can have multiple artifacts which hinder their usability in the downstream segmentation task. Exist more simple approach like PØDA strategy [39] which is more favorable than CLIPstyler for downstream tasks like semantic segmentation: the minimal statistics changes help avoiding significant drifts on the feature manifold which may result in unwanted errors. The PODA solution attempts to mitigate the domain-shift problem of the unsupervised domain adaptation framework (UDA).

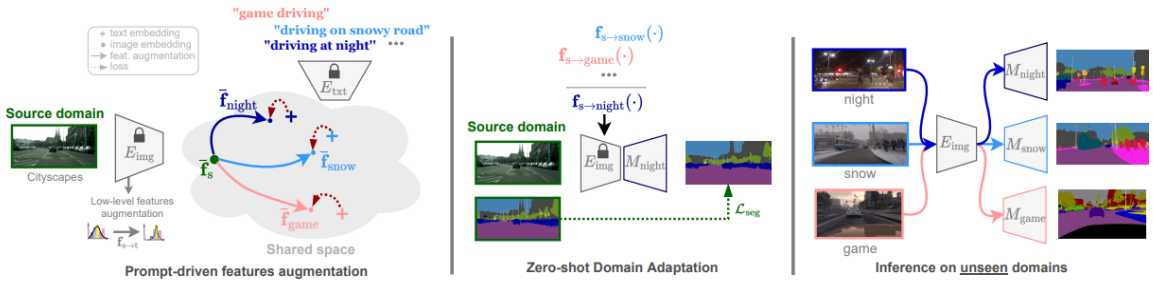

Fig. 30: PØDA: Prompt-driven Zero-shot Domain Adaptation. (Left) Using only a single textual description of an unseen target domain, it leverage a frozen ResNet encoder with CLIP weights to learn source-target low-level features stylizations over the source image embedding that bring the augmented embeddings closer to their respective target prompt embeddings. [39]

Finally, these models like ClipStyler lack the ability to mimic the appearance of subjects in a given reference set and synthesize novel renditions of them in different contexts. It is possible to "personalize" with text-to-image diffusion models.

With just a few images (typically 3-5) of a subject, DreamBooth [40] can generate a myriad of images of the subject in different contexts, using the guidance of a text prompt.

The key idea of DreamBooth is to embed a given subject instance in the output domain of a text-to-image diffusion model by binding the subject to a unique identifier.The fine-tuning process can work given only 3-5 subject images, making the technique particularly accessible. Exist method which differ from existing text-based image editing works like Dream Booth in that it enables editing from instructions that tell the model what action to perform. A key benefit of following editing instructions is that the user can just tell the model exactly what to do in natural written text.

This kind of method editing images from human instructions too : given an input image and a written instruction that tells the model what to do, the model follows these intructions to edit the image. To obtain training data for this problem, combine the knowledge of two large pretrained models—a language model (GPT-3) and a text-to-image model (Stable Diffusion)—to generate a large dataset of image editing examples. The conditional diffusion model, InstructPix2Pix [41] is trained on generated data, and generalizes to real images and user-written instructions at inference time. Since it performs edits in the forward pass and does not require per-example fine-tuning or inversion, the model edits images quickly, in a matter of seconds.

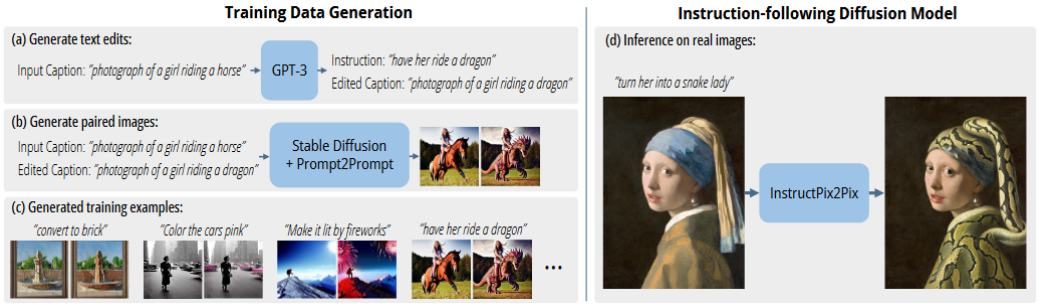

Fig. 31: Pix2pix overview [41]

The method consists of two parts: generating an image editing dataset, and training a diffusion model on that dataset. (a) finetuned GPT-3 is used to generate instructions and edited captions. (b) then StableDiffusion in combination with Prompt-to-Prompt to generate pairs of mages from pairs of captions. Thanks to this procedure,a dataset is created of over 450,000 training examples. Finally,the InstructPix2Pix diffusion model is trained on generated data to edit images from instructions.

6.2 Instruct Pix2Pix strategy presentation

Instruct Pix2Pix is an approach that combines two large pretrained models, a large language model and a text-to- image model, to generate a dataset for training a diffusion model to follow written image editing instructions. This technique now represents the state of the art in image editing instructions. Moreover, the mthod is able to produce a wide variety of compelling edits to images, including style, medium, and other contex- tual changes and there still remain a number of limitations

For these reasons, this strategy was chosen for augmenting RUGD. The idea is therefore to use the Instruct Pix2Pix model, which has already been trained, and to run the inferences on the RUGD data, for each image and n times, with n corresponding to the number of images.

5 weather augmentations were planned beforehand:

- Winter atmosphere with the prompt "I want to have a lot of snow in the ground and a fall of snow".
- Flooded grounded" with the prompt "I want the ground to be flooded".
- Light fog with the prompt "Add a very light fog".
- High grass with the prompt "I want the grass to be higher".
- No leaves in the trees with the prompt "I want the trees to have no leaves".

It is important to notice that exist Classifier-free guidance weights over two conditional inputs. w1 controls similarity with the input image, while w2 controls consistency with the edit instruction. Increasing weight text results in a stronger edit applied to the image (i.e., the output agrees more with the instruction), and increasing sI can help preserve the spatial structure of the input image (i.e.,the output agrees more with the input image). We find that values of sT in the range 5-10 and values of sI in the range 1-1.5 typically produce the best results according to the paper.

The prompts were not chosen at random. They were first tested from the test interface hosted by hugging Face [42]. Several tests were carried out to obtain the best possible quality.
After launching the tests of these augmentations with default weights, three problems were quickly identified:

- In the case of the RUGD images of the "creek" environment (an environment where the density of vegetation is fairly dense, images taken in the middle of a forest), the augmentation in data was barely visible, thus skewing the data augmentation produced. Generally speaking, the data augmentation produced in all the environments was irregular, with sometimes very realistic augmentations and other times where the augmentation was barely visible. This phenomenon was much more pronounced in the creek environment. To counter this phenomenon, conditions have been added to the RUGD data inference launch, for example an augmentation in text weight via an if condition coded directly in the launch algorithm.
- For the "no leaves in the trees" data augmentation, even when attempting to vary the weights, the data augmentations obtained were not realistic enough and even distorted the semantic structure of the image if we keep the weights by default. Example of output obtained can be seen in the appendix A.12 .It was therefore decided not to retain this data augmentation for this reason.

- Each inference for a given data augmentation takes 1.5 days. In order to optimize the time remaining for the final experiments, it was decided to carry out 4 data augmentations instead of 3. The "grass higher" data augmentation was therefore not taken into account.

This data augmentation strategy can be summarised as follows in the figure 32 :

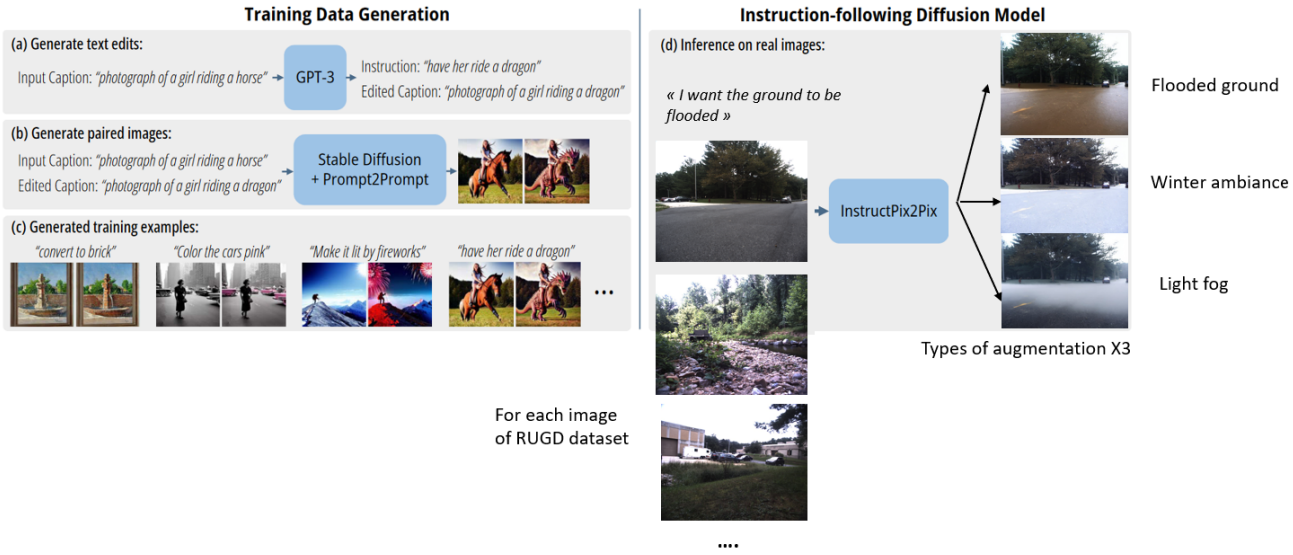

Fig. 32: Pix2pix overview applied to RUGD data augmentation case

6.3 Evaluation DAFormer RUGD with nominal conditions to RUGD with adverse conditions

The data augmentation of the RUGD data was a key stage in the internship, enabling the DAFormer to begin experimenting with this augmented data.
The RUGD augmented data has been adapted and organised strictly in the same way as the RUGD data, with the same classes and the same environments distributed in the same train, test and validation splits.
It is also essential to point out that the new augmented data are not labelled with the augmentations in mind. Their associated labels are the same as those of the RUGD dataset.
All the results obtained are summarized in the table 5 below:

Méthode	MIou on train set (%)	MIou on test set (%)
Source only RUGD augmented	19.12	16.85
DAFormer RUGD to RUGD augmented	28.11	23.0
Oracle RUGD augmented	33.11	29.78

Tab. 5: Barometer results of the DAFormer model for RUGD to RUGD augmented

In addition, the appendix A.14 contains all the qualitative results (prediction masks) from these experiments.
For both experiments, the oracle results remain superior to the UDA results. And the latter exceed those of source-only. So we're in the barometer's ideal case too. We reach a "UDA gain" of

$$\frac{53.59 - 43.81}{76.26 - 43.81}$$

equals to 47 %. This shows that the UDA, through the DAFormer, makes it possible to gain in performance on the semantic segmentation task compared to source-only.
From the class point of view, all results are shown in the appendix A.13.
Generally speaking, the equivalent performance of source only and DAFormer in terms of metrics is confirmed qualitatively with the prediction masks obtained. Visually, the differences in prediction are often minimal (see the example in Fig 52). However, we note that the classes predictions not modified by data augmentation perform very well, and their metrics reach the same scores as those achieved by the supervised oracle. Examples include the "sky" and "tree" classes, which very often have high scores of over 60% MIoU and all the visualization predictions prove it.
Conversely, the classes predictions of DAFormer modified by the data augmentation perform less well to the predictions of Oracle, which can sometimes explain the discrepancies in prediction that are visible visually but also in the metric scores per class. We can quote two examples :

- Compared with the ground truth, the gravel zone in the bottom right of the figure 53 is completely well predicted by the supervised oracle. On the other hand, the DAFormer does not correctly predict the entire gravel area, with a portion that it predicts as "grass".
- The same case can be seen in Fig 49 . where, on the left of the image, the DAFormer confuses bush and tree leaves by predicting the "bush" class for certain pixels instead of the "tree" class. At this precise level of the image, the area of poor prediction is greater for that predicted by the DAFormer.

In the worst case, some classes may not even be detected because of data augmentation, as is the case in the figure 50, where the fog disturbs the model, which does not predict the good classes. Disturbed by the fog, it predicts a section of the road to be an area of water in places.

6.4 Focus on the Samba dataset experimentations

Previous experiments have shown the potential gains that unsupervised domain adaptation could bring in bridging the gap between RUGD and augmented RUGD.
The second issue is whether these experiments can be transposed to Samba data, Safran's private data. They can only be transposed if the Samba dataset is semantically close to the RUGD dataset with a low domain shift.

A first simple solution is to learn more about the Samba data by applying several segmentation models to it: from the classic segformer to the RUGD to Samba DAFormer. The idea is to collect these results, compare them with those obtained for RUGD and provide interpretations.

First of all, I had to think about mapping the RUGD classes to Samba in order to apply the DAFormer model. I had several choices: to use the initial 25 RUGD classes or to use the existing mapping of the 9 meta-classes defined in the table 29. In order to avoid doing a class mapping on an existing mapping class, I decided to choose the first solution. This choice can be criticized in the last chapter of the report.

The semantic data of Samba by image being classified as "navigable" or "non-navigable", I carried out the mapping of the classes in close collaboration with the autonomous vehicle experts of the CASPer team who helped me with the mapping. The definition of the Samba mapping classes are degined in the following figure 33.

Méta classes	Navigable	Non Navigable
Initial classes	Dirt Sand Grass Asphalt Gravel Mulch Rock-bed	Void, Tree, Pole, Water Sky, Vehicle, Container Building, Log, Bicycle Person, Fence, Bush Sign, Rock, Bridge Concrete, Picnic table

Fig. 33: Class mapping overview table for Samba classes

All the results obtained are summarized in the table 6 below:

Méthode	MIou on train set (%)	MIou on test set (%)
Source only Samba	80.19	76.08
DAFormer RUGD to Samba	85.83	84.17
Oracle Samba	87.57	86.03

Tab. 6: Barometer results of the DAFormer model for RUGD to Samba

From the class point of view, all results are shown in the appendix A.15. In addition, the appendix A.16 contains all the qualitative results (prediction masks) from these experiments. From the predictive masks, we can visually see that the oracle is able to predict the 'navigable' class in the image slightly better, with a slightly higher level of accuracy. For example, in the second figure of the appendix, the DAFormer is unable to predict the "navigable" class at the bottom of the image when the path continues at the bottom of the forest. However, generally speaking, the predictive masks between the two models (DAFormer and Oracle) are almost equivalent, proving the very good performance of the UDA model.

For both experiments, the oracle results remain slightly superior to the UDA results. And the latter exceed those of source-only. So we're in the barometer's ideal case too. We reach

a "UDA gain" of

$$\frac{84.17 - 76.08}{86.06 - 76.08}$$

equals to 81 %.

From metrics point of view, we note the high MIoU metric performance of the DAFormer model reaching 80.72%, much higher than the performance of DAFormer from cityscapes to RUGD and from cityscapes to ACDC seen previously. Same observation for the "UDA gain". As we can see in the table 7, the DAFormer gain for the RUGD to Samba case is the highest. This confirms the qualitative results observed in the appendix.

Use case	UDA - source-only (%)	Oracle - Source only (%)	DAFormer Gain(%)
Cityscapes2RUGD	3.74	12.06	33
Cityscapes2acdc	9.78	32.45	30
RUGD2RUGDaugmented	6.15	12.93	23.0
RUGD2Samba	8.09	9.98	81

Tab. 7: Comparaison results between DaFormer experimentations

A more detailed interpretation of the above results will be given in the next chapter of this report.

7 Discussion and perspectives

The results presented above are the outcome of a 6-month research internship. To address the issues set out at the beginning of the report, I followed a scientific approach involving experimentation on a variety of data, using a state-of-the-art, easily scalable UDA architecture and successive iterations with my internship tutor and the CasPer team to bring the actions to fruition. All these choices were justified and are now the result of my 6-month internship. In order to put this work into perspective, this chapter summarizes the main areas for improvement and discussion in order to take this work even further and go beyond the initial research problem.

7.1 Performance improvements

The performances obtained must be judged according to the associated results but also according to their representativeness, asking whether they are representative and allow us to respond to the initial problem. These two aspects can be studied from two points of view :

7.1.1 From the data augmentation point of view

Data augmentation of RUGD data was carried out using the Instruct Pix2Pix method. The latter made it possible to launch a first experiment with the DAFormer from RUGD to RUGD

augmented which gave rise to very satisfactory results. Indeed, the evaluation metric almost reaches that of the supervised oracle.

However, thisis the result of artificial data augmentation. The water on the ground in RUGD's images is artificial, not real just like the fog. In addition, as explained in the previous chapter, the data augmentation provided on RUGD is irregular depending on the environment in which the image was captured despite standardization efforts.

These results are not specific to my experiments since several limitations to this strategy have been listed in the research paper [43]. indeed, the model is limited by the visual quality of the generated dataset, and therefore by the diffusion model used to generate the imagery (in this case, Stable Diffusion [52]). Furthermore, our method's ability to generalize to new edits and make correct associations between visual changes and text instructions is limited by the human-written instructions used to fine-tune GPT-3, by the ability of GPT-3 to create instructions and modify captions, and by the ability of Prompt-to-Prompt [17] to modify generated images. In particular, the model struggles with spatial reasoning (e.g., "add fog to the left of the image"), just as in Stable Diffusion and Prompt-to-Prompt. So, we do not control the location of the data augmentation provided on the image. This can cause recurring biases in the results obtained. With, for example, data augmentation located precisely on a particular class of the dataset (eg fog on a vehicle) which biases the prediction of the class in relation to the ground truth.

It would therefore have been interesting not only to apply the inference mechanism for our use case but to dive more into the training mechanism of the diffusion model and the image/caption matching mechanism by Stable diffusion. However, this is a much larger subject which requires a considerable amount of time to agree on. The work also opens up questions, such as how to follow instructions for spatial reasoning, how to combine instructions with other conditioning modalities like user interaction, and how to evaluate instruction-based image editing.

It would therefore be interesting to re-test our RUGD DAFormer on augmented RUGD taking into account a much more robust data augmentation.

7.1.2 From the model point of view

At the level of the unsupervised domain adaptation model, the state-of-the-art DAFormer architecture was chosen. One challenge of training a more capable architecture for UDA is overfitting to the source domain. To circumvent this issue, we introduced at the begining of the report three strategies to stabilize and regularize the UDA training: Rare Class Sampling, Thing-Class ImageNet Feature Distance, and learning rate warmup. In their implementation, these strategies use hyper-parameters which were defined identically in the experiments of this internship and which can be briefly recalled :

For the DAFormer architecture, the MiT-B3 encoder, which produces a feature pyramid with $C = [64, 128, 320]$ was used. The DAFormer decoder uses $C_e = 256$ and dilation rates of

1 , 6,12 , and 18. All encoders are pretrained on ImageNet-1k. Training In accordance with [47, 87], I trained DAFormer with AdamW [44], a learning rate of $\eta_{\text{base}} = 6 \times 10^{-5}$ for the encoder and 6×10^{-4} for the decoder, a weight decay of 0.01 , linear learning rate warmup with $t_{\text{warm}} = 1.5\text{k}$, and linear decay afterwards. It is trained on a batch of two 512×512 random crops for 40k iterations. Following DACS [22], I use the same data augmentation parameters as defined in the paper and set $\alpha = 0.99$ and $\tau = 0.968$. The RCS temperature is set $T = 0.01$ to maximize the sampled pixels of the class with the least pixels.

Furthermore, rare class sampling (RCS) augmentations the evaluation metrics by around 4 to 5% in the case of DAFormer taking account it. For example below, with the ablation study with and without rcs for the application of RUGD's DAFormer to Samba :

Méthode	MIou on train set (%)	MIou on test set (%)
DAFormer RUGD to Samba with RCS	85.83	84.17
DAFormer RUGD to Samba without RCS	83.13	80.12

Tab. 8: Results experimentations of DAFormer model with and without RCS

For Feature Distance (FD), $r = 0.75$ and $\lambda_{FD} = 0.005$ to induce a similar gradient magnitude into the encoder as $\mathcal{L}_S$.

In this context, the best UDA mIoU is achieved by the MiT-B5 encoder instead of MiT-B3. It would have been interisting to test it. We have seen previously that the proposed RCS augmentations the sampling probability of rare classes in order to avoid underrepresentation of some classes during the training.The temperature T controls the smoothness of the distribution. The temperature T is chosen to reach a balance of the number of re-sampled pixels of classes with small and medium frequency by maximizing the number of re-sampled pixels of the class with the least. It would have been interesting to test different values of the tempeture to detect or not the influence in the score metrics.

While RCS gives a performance boost, the performance for thing-classes (like vehicle class for example) could still be further improved as some of the object classes that are fairly well separated in ImageNet features are mixed together after the UDA training. It would have been interesting to test the DAFormer without the Features Distance (FD) technique in order to show its influence in the performance and to prove so applying FD only to thing-classes, which the ImageNet features were trained on, is important for its good performance as demonstrated in the DAFormer paper.

Finally, a common problem in the UDA methods like the DAFormer model is the confusion of classes with a similar visual appearance on the target domain such as road/sidewalk or pedestrian/rider as there is no ground truth supervision available to learn the slight appearance differences. For example, the interior of the sidewalk in Figure 34 is segmented as road, probably, due to a similar local appearance. To address this problem, enhance UDA is possible with spatial context relations as additional clues for robust visual recognition.

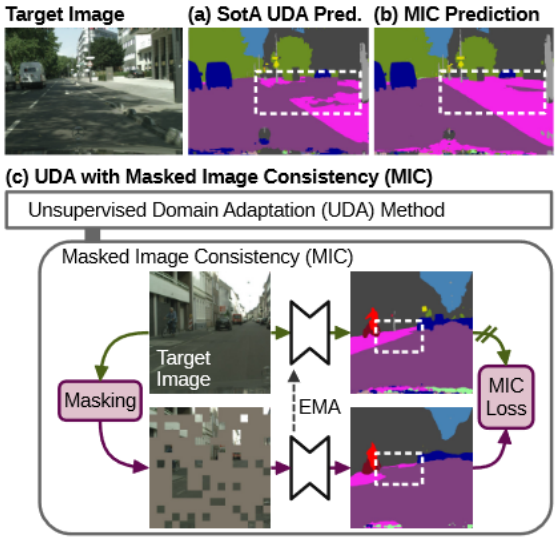

Fig. 34: (a) Previous UDA methods such as DAFormer struggle with similarly looking classes on the unlabeled target domain.Here, the interior of the sidewalk is wrongly segmented as road, probably, due to the ambiguous local appearance. (b) The proposed Masked Image Consistency (MIC) enhances the learning of context relations.

Although the used network architectures already have the capability to model context relations, previous UDA methods are still not able to reach the full potential of using context dependencies on the target domain as the used unsupervised target losses are not powerful enough to, enable effective learning of such information. Therefore, a novel Masked Image Consistency (MIC) [29], whose paper was accepted by the CVPR 2023, plug-in for UDA was proposed which can be applied to various visual recognition tasks. Considering semantic segmentation for illustration, MIC masks out a random selection of target image patches and trains the network to predict the semantic segmentation result of the entire image including the masked-out parts. In that way, the network has to utilize the context to infer the semantics of the masked regions. As there are no ground truth labels for the target domain, the pseudo-labels are used again, generated by an EMA teacher that uses the original, unmasked target images as input. The only difference is that the teacher can utilize both context and local clues to generate robust pseudo-labels.

It sets the most recent state-of-the-art performance on all tested benchmarks with significant improvements over previous methods like DAFormer. The response to the internship problem is firstly based on a single state-of-the-art architecture from the UDA, namely the DAFormer for which several experiments have been carried out. It would have been relevant, with more time, to compare the performance of the DAFormer with the MIC architecture.

7.2 Focus on the domain shift RUGD to Samba

The results of the DaFormer barometer from RUGD to Samba were obtained in the previous chapter and demonstrate an excellent performance of the UDA. The results obtained could

have been better if the class mapping to Samba had been done over the existing class mapping used for the generalization on Cityscapes as defined in the fig 29. It would have been interesting to compare the two scenarios and to be able to illustrate before and after class mapping, the prediction boundary between classes using the T-SNE visualization tool.
The results of the RUGD to Samba DAFormer barometer highlights three important aspects:

- the excellent MIoU of metric obtained by the DAFormer reaching 84.17 % on the test data according to the table 6. This is also the best performance among all the DAFormer experiments carried out during the internship.
- The gap between DAFormer of RUGD to Samba and the MIoU scores of source only is the second hightest among all the DAFormer tests according to the table 7 with 8.09%. The DAFormer method therefore manages to fill (by 8%) the lack of generalization of the source-only method. This is also transcribed in the table 7 with the DAFormer gain metric which is by far the highest for the case of RUGD to Samba. This proves that the gap between UDA and source-only has been greatly reduced in terms of performance and that UDA in this case brings real added value.

  By putting in perspective this very good result with the other experiments of cityscapes towards RUGD or Cityscapes towards ACDC, a deduction on the domain shift can be made. The gain of UDA for the case RUGD to Samba being so important and superior to the other gains of the other experiments of UDA, that one can deduce from it that at the beginning, the initial gap between the two datasets is minimal and that finally the task of UDA was not very complicated to carry out.
- The gap between the oracle MIoU scores and the source-only MIou score is the lowest among all the DAFormer tests according to the table 7 with 9.98%.

These results must be interpreted in their entirety. It suggest that the domain shift, the gap between RUGD and Samba is not very important as we thought it at the start of the internship based solely on the difference in semantic information between the two datasets. We can note that this is the completely opposite results and interpretations for the DAFormer of Cityscapes to RUGD.

However, these interpretations are based on experiments and prediction metrics and not on metrics specific to the evaluation of domain discrepancy between two domains.
Fortunately, exist metrics that measure the distribution discrepancy between the source and target domains to indicate model performance, as these metrics represent feature transferability. We can consider the Maximum Mean Discrepancy (MMD) metric that we have seen in the state of the art part of this report. Exist others metrics CORAL [14], the Kullback-Leibler divergence [45] and the Wasserstein Distance [46] from a label shift point of view but the MMD-based methods are widely used. To further support our interpretations of the weak domain shift between RUGD and Samba, it would have been interesting to implement and test one of these technique.

7.3 Catastrophe forgetting

The creation of large-scale open domain reading comprehension data sets in recent years has enabled the development of end-to-end neural comprehension models with promising results. To use these models for domains with limited training data, one of the most effective approach is to first pre-train them on large out-of-domain source data and then fine-tune them with the limited target data. The caveat of this is that after fine-tuning the comprehension models tend to perform poorly in the source domain, a phenomenon known as catastrophic forgetting.

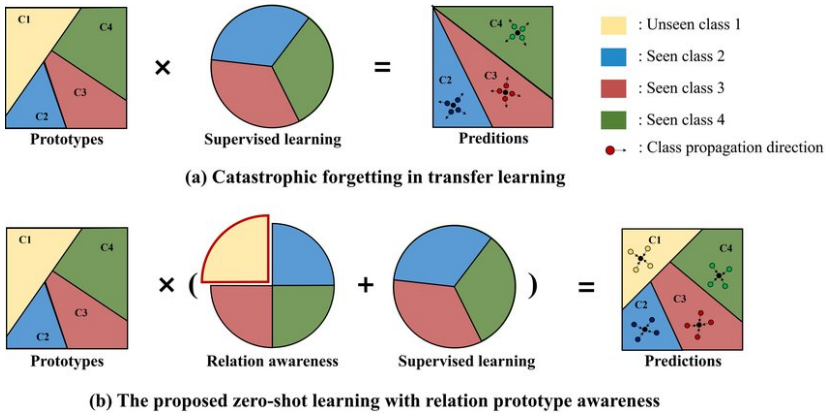

Fig. 35: Catastrophic forgeting overview in the global context of Transfer learning

We can retrieve this catastrophic forgetting in the context of unsupervised domain adaptation. The goal is to preserve the source domain's performance as much as possible, while keeping target domain's performance optimal and assuming no access to the source data. Thus, it is necessary to calculate the performance on the source dataset after adaptation and check that the metrics have not decreased.

8 Conclusion

The internship was an opportunity to focus on the typical case of Clear weather to Adverse Weather and to discover if Unsupervised Domain Adaptation is suitable for this type of task through experiments on a private Safran dataset called Samba. With few semantic classes, the Samba dataset was not representative of an offroad dataset in conditions with a large variety of classes, which posed a problem for our tests. A proxy for this dataset was therefore chosen, in this case the RUGD dataset on which the UDA experiments were carried out.
As well as evaluating our UDA model from RUGD to an augmented RUGD (data augmentation carried out to generate difficult weather conditions), it was also important to prove through results metrics that RUGD is a good proxy for Samba, i.e. that the domain shift between the two datasets is so small that the results and interpretations made on RUGD can be transposed to Samba.

As we can see in the appendix A.17, the results obtained by our UDA model on the augmented RUGD to RUGD task are close to those obtained by the oracle in the supervised

framework and are superior to the source-only results. Positioned in the ideal section of our evaluation barometer, the score obtained proves that the domain adaptation task in difficult weather conditions can at least constitute a robust solution in the case where the target domain data is not labelled. However, this interpretation needs to be weighed against the fact that the data augmentation carried out by the Instruct Pix2Pix method is artificial, not real, and also not uniform for all the images in the dataset.

Furthermore, it is important to notice this data augmentation task in the context of unsupervised domain adaptation can be compared by the general Domain Adaptation framework called Domain generalization [25]. Indeed, to counter the lack of knowledge about the target domain, several strategies are possible whose data augmentation. In our case, our data augmentation is , in a way, conditioned by the style of weather that we want brought in the image.

These first experiments were carried out on RUGD, the state-of-the-art off-road dataset for autonomous vehicles. However, the CASPer team's strategic need is to know whether these experiments can be transposed to SAMBA data and also to know whether the results obtained can be valid in the Safra context. By putting in perspective all of our experiments of our UDA models, we can therefore affirm that RUGD dataset is a good proxy of the Samba dataset but that it would nevertheless be interesting to exploit specifical metrics to evaluate with forecast this weak domain shift and the data discrepancy .

To conclude, in the short term, the challenge for the CASPer team would be to be able to apply all of the experiments on the Samba dataset. The application on Samba will however pose two problems:

- It will be necessary to carry out a data augmentation of Samba data, in the same way as that carried out on RUGD but in more robust. Or, depending on Safran's budget, to carry out data acquisition campaigns with the autonomous vehicle in winter in order to better exploit difficult weather conditions.
- the need to extend the "navigable" and "non-navigable" classes to the same classes of RUGD describing the understanding of the scene and the environment. This would require automating image labeling with automatic class mapping. Today in the state of the art, the recent solution would be SAM [47] (Segment anything Model) which automatically segment everything in an image.

This internship, which initially focused on the task of unsupervised domain adaptation and its evaluation in the context of difficult weather conditions, ultimately raised several questions on related subjects such as data augmentation mechanisms, on the evaluation of the domain shift between two data sets as well as on the border between the UDA framework and data generalization. This made the research internship even more interesting and enriched my knowledge on these subjects.

A Appendicies

A.1 Samba dataset

The environment dataset Samba has the following characteristics:

- Forests
- Disused buildings (bunkers)
- Railway line
- Moderately deteriorated asphalted streets and lanes
- Unmarked carriageways
- Presence of ruts and potholes.

The data was acquired on a single sunny day. As a result, the images are very bright. The data collected consists of :

- Camera video stream: 6 video sequences with a total of 30157 frames
- Sparse depth maps produced by projecting the LIDAR data onto the image frame.

An example of the images making up the database is shown below :

Fig. 36: Overview of the Samba dataset

	Output Size	Layer Name	Mix Transformer					
			B0	B1	B2	B3	B4	B5
Stage 1	$\frac{H}{4} \times \frac{W}{4}$	Overlapping Patch Embedding	$K_1 = 7;\ S_1 = 4;\ P_1 = 3$					
			$C_1 = 32$	$C_1 = 64$				
		Transformer Encoder	$R_1 = 8$ $N_1 = 1$ $E_1 = 8$ $L_1 = 2$	$R_1 = 8$ $N_1 = 1$ $E_1 = 8$ $L_1 = 2$	$R_1 = 8$ $N_1 = 1$ $E_1 = 8$ $L_1 = 3$	$R_1 = 8$ $N_1 = 1$ $E_1 = 8$ $L_1 = 3$	$R_1 = 8$ $N_1 = 1$ $E_1 = 8$ $L_1 = 3$	$R_1 = 8$ $N_1 = 1$ $E_1 = 4$ $L_1 = 3$
Stage 2	$\frac{H}{8} \times \frac{W}{8}$	Overlapping Patch Embedding	$K_2 = 3;\ S_2 = 2;\ P_2 = 1$					
			$C_2 = 64$	$C_2 = 128$				
		Transformer Encoder	$R_2 = 4$ $N_2 = 2$ $E_2 = 8$ $L_2 = 2$	$R_2 = 4$ $N_2 = 2$ $E_2 = 8$ $L_2 = 2$	$R_2 = 4$ $N_2 = 2$ $E_2 = 8$ $L_2 = 3$	$R_2 = 4$ $N_2 = 2$ $E_2 = 8$ $L_2 = 3$	$R_2 = 4$ $N_2 = 2$ $E_2 = 8$ $L_2 = 8$	$R_2 = 4$ $N_2 = 2$ $E_2 = 4$ $L_2 = 6$
Stage 3	$\frac{H}{16} \times \frac{W}{16}$	Overlapping Patch Embedding	$K_3 = 3;\ S_3 = 2;\ P_3 = 1$					
			$C_3 = 160$	$C_3 = 320$				
		Transformer Encoder	$R_3 = 2$ $N_3 = 5$ $E_3 = 4$ $L_3 = 2$	$R_3 = 2$ $N_3 = 5$ $E_3 = 4$ $L_3 = 2$	$R_3 = 2$ $N_3 = 5$ $E_3 = 4$ $L_3 = 6$	$R_3 = 2$ $N_3 = 5$ $E_3 = 4$ $L_3 = 18$	$R_3 = 2$ $N_3 = 5$ $E_3 = 4$ $L_3 = 27$	$R_3 = 2$ $N_3 = 5$ $E_3 = 4$ $L_3 = 40$
Stage 4	$\frac{H}{32} \times \frac{W}{32}$	Overlapping Patch Embedding	$K_4 = 3;\ S_4 = 2;\ P_4 = 1$					
			$C_4 = 256$	$C_4 = 512$				
		Transformer Encoder	$R_4 = 1$ $N_4 = 8$ $E_4 = 4$ $L_4 = 2$	$R_4 = 1$ $N_4 = 8$ $E_4 = 4$ $L_4 = 2$	$R_4 = 1$ $N_4 = 8$ $E_4 = 4$ $L_4 = 3$	$R_4 = 1$ $N_4 = 8$ $E_4 = 4$ $L_4 = 3$	$R_4 = 1$ $N_4 = 8$ $E_4 = 4$ $L_4 = 3$	$R_4 = 1$ $N_4 = 8$ $E_4 = 4$ $L_4 = 3$

Fig. 37

A.2 Mit series

In summary, the hyper-parameters of our MiT are listed as follows:

- K_i : the patch size of the overlapping patch embedding in Stage i; - S_i : the stride of the overlapping patch embedding in Stage i;

- P_i : the padding size of the overlapping patch embedding in Stage i;

- C_i : the channel number of the output of Stage i;

- L_i : the number of encoder layers in Stage i;

- R_i : the reduction ratio of the Efficient Self-Attention in Stage i;

- N_i : the head number of the Efficient Self-Attention in Stage i;

- E_i : the expansion ratio of the feed-forward layer [78] in Stage i;

Table shows the detailed information of our MiT series with H the height and W the width of a given image. To facilitate efficient discussion, we assign the code name B0 to B5 for MiT encoder, where B0 is the smallest model designed for real-time, while B5 is the largest model designed for high performance.

A.3 Cityscapes dataset

Cityscapes dataset is a new large-scale dataset that contains a diverse set of stereo video sequences recorded in street scenes from 50 different cities, with high quality pixel-level annotations of 5000 frames in addition to a larger set of 20000 weakly annotated frames. The dataset is thus an order of magnitude larger than similar previous attempts. Details on annotated classes and examples of our annotations are available at this webpage [35].

The Cityscapes Dataset is intended for assessing the performance of vision algorithms for major tasks of semantic urban scene understanding: pixel-level, instance-level, and panoptic semantic labeling; supporting research that aims to exploit large volumes of (weakly) annotated data, e.g. for training deep neural networks.

Retrieved from the official Cityscapes dataset website, below a few characteristics of the data :

- Polygonal annotations : Dense semantic segmentation

- Complexity : 19 classes, Labeled foreground objects must never have holes, i.e. if there is some background visible 'through' some foreground object, it is considered to be part of the foreground. This also applies to regions that are highly mixed with two or more classes: they are labeled with the foreground class. Examples: tree leaves in front of house or sky (everything tree), transparent car windows (everything car).

- Diversity : 50 cities, several months (spring, summer, fall), daytime, good/medium weather conditions, manually selected frames, large number of dynamic objects, varying scene layout, Varying background

- Volume : 5000 annotated images with fine annotations

Metadata : preceding and trailing video frames. Each annotated image is the 20th image from a 30 frame video snippets, corresponding right stereo viewx, GPS coordinates, Egomotion data from vehicle odometry.

Fig. 38: Examples of the high quality dense pixel annotation in an cityscapes image from Stuggart city

A.4 RUGD Dataset

The RUGD dataset focuses on semantic understanding of unstructured outdoor environments for applications in off-road autonomous navigation. The dataset is comprised of video sequences captured from the camera onboard a mobile robot platform. The overall goal of the data collection is to provide a more representative dataset of environments that lack structural cues that are commonly found in urban city autonomous navigation datasets. The platform used for data collection is rugged enough to traverse through challenging terrain to explore more unstructured areas of an environment. Dense pixel-wise annotations are provided for every fifth frame in a video sequence. The ontology is defined to support fine-grained terrain identification for path planning tasks, and object identification to avoid obstacles and localize landmarks. In total, 24 semantic categories (figure 39) can be found in the annotations of the videos including areas that represent four general environment categories:

- creek - areas near a body of water with some vegetation
- park - woodsy areas with buildings and paved roads
- trail - areas representing non-paved, gravel terrain in woods
- village - areas with buildings and limited paved roads

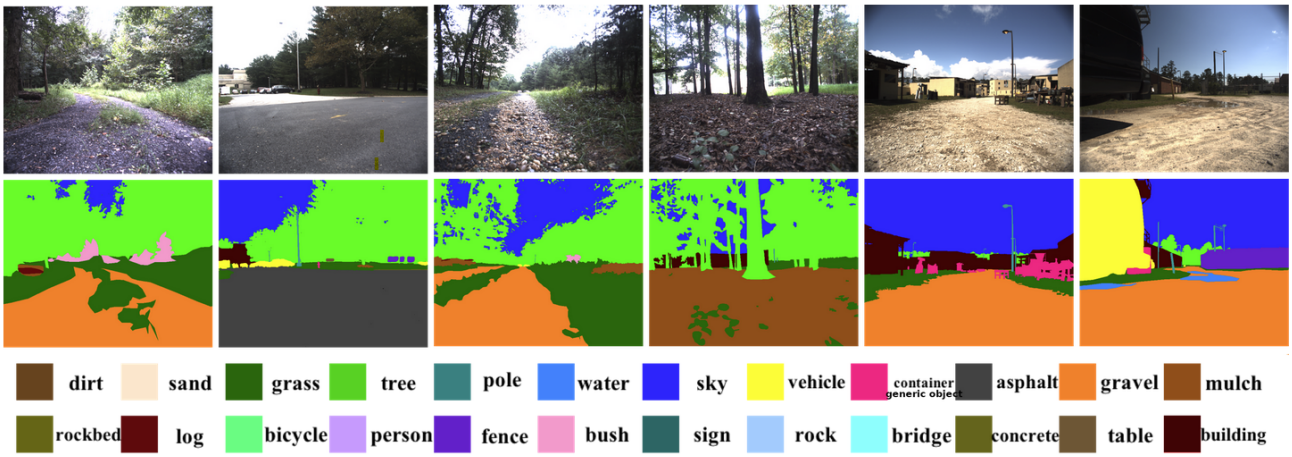

Fig. 39: RUGD classes overview

Videos from the RUGD dataset are partitioned into train/val/test splits for our benchmark experimental evaluation. Figure 40 lists which videos belong to each data split, with 64% of the total annotated frames used for training, 10% for validation, and the remaining 26% for testing. While the selection of videos for each split was decided to roughly produce specific sizes of each split, two videos were specifically placed in the test split to test realistic challenges faced in many motivating applications. First, the creek sequence is the only example with significant rock bed terrain. Reserving this as a testing video demonstrates how existing architectures are able to learn from sparse label instances. This is a highly realistic

scenario in unstructured environment applications as it is difficult to collect large amounts of training data for all possible terrain a priori. Second, the train-7 sequence represents significantly more off-road jitter than others, producing many frames that appear to be quite blurry. This property is also present in training sequences, but again we reserve the difficult video to determine how well the techniques are able to perform under these harsh conditions.

	C	P1	P2	P8	T	T3	T4	T5	T6	T7	T9	T10	T11	T12	T13	T14	T15	V	Total	%
train			x		x	x	x		x		x	x	x	x		x	x	x	4759	63.83
val				x				x											733	9.83
test	x	x								x					x				1964	26.34

Fig. 40: test

A.5 Segformer on RUGD - Quantitative results per class

Class	IoU on test set (%)	Acc (%)
void	0.003	0.003
dirt	0.74	0.8
sand	45.06	49.77
grass	74.26	93.12
tree	82.28	96.91
pole	82.28	96.91
water	36.69	44.14
sky	29.53	29.9
vehicle	85.91	93.44
container	57.49	84.05
asphalt	55.32	8.3
gravel	29.94	53.55
building	30.67	92.85
mulch	72.19	87.23
rock-bed	61.6	91.2
log	21.57	21.68
bicycle	51.16	63.58
person	0.0	nan
fence	0.0	0.0
bush	63.47	78.08
sign	19.85	20.5
rock	13.23	14.13
bridge	20.59	34.07
concrete	0.0	0.0
picnic table	88.81	96.2

Tab. 9: Results of the barometer per class of the Segformer model on RUGD dataset

A.6 segformer on Cityscapes - Qualitative results

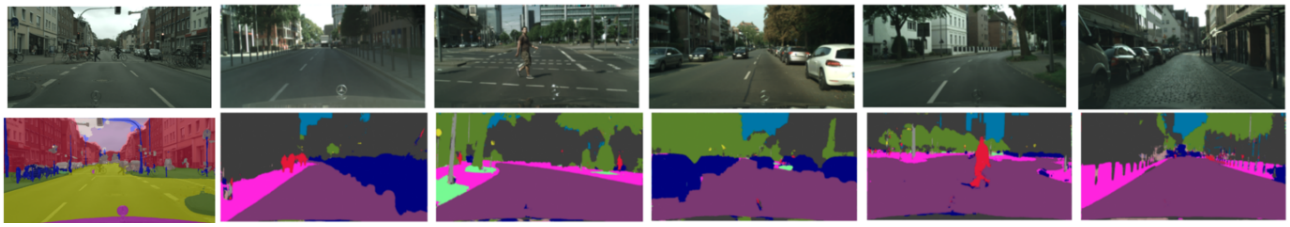

Fig. 41: test

A.7 Segformer on RUGD - Quanlitative results

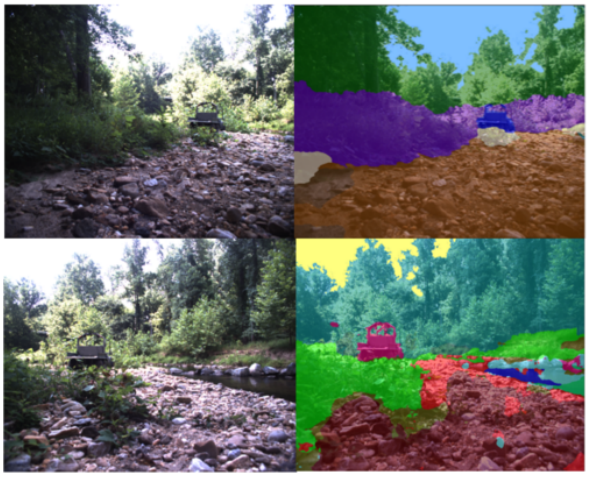

Fig. 42

A.8 Segformer on RUGD meta classes - Qualitative results

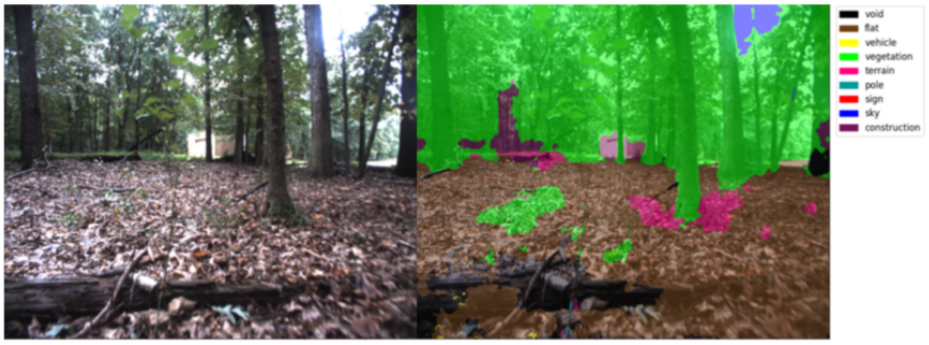

Fig. 43

A.9 Barometer DAFormer Cityscapes to RUGD with IoU test results by meta-classes

Class	Source only	DAFormer	Oracle
void	0.003	1.15	14.59
flat	5.5	8.74	21.92
vehicle	11.82	13.89	32.63
vegetation	42.11	75.67	74.87
terrain	9.34	10.35	44.75
pole	18.75	21.35	32.67
sign	0.6	1.04	4.49
sky	22.87	35.18	35.2
construction	9.8	11.46	14.53

Tab. 10: Results per class of the Segformer model on RUGD dataset

A.10 DAFormer Cityscapes to RUGD - Qualitative results on meta-classes

Fig. 44

A.11 DAFormer Cityscapes to ACDC - Qualitative results on meta-classes

Fig. 45

A.12 Instruct Pix2Pix Output obtained with "no leaves in trees" prompt

Fig. 46: In the left, the original image, in the middle augmented image with "no leaves in trees" (image weight : 7.5, text weight : 1.5), in the right augmented image with "no leaves in trees" (image weight : 7.5, text weight : 4.5)

A.13 Barometer DAFormer RUGD to RUGD augmented with IoU test results by class - Quantitative result

Class	Source only	DAFormer	Oracle
void	0	0	0
dirt	0	0.03	0.51
sand	0	0	0
grass	54.26	58.18	62.19
tree	74.28	83.96	84.2
pole	2.01	2.06	2.06
water	31.69	34.2	36.2
sky	30.12	46.56	48.29
vehicle	32.45	41.09	56.09
container	1.67	2.73	10.2
asphalt	55.32	62.8	61.23
gravel	29.94	42.81	53.51
building	10.61	23.75	30.67
mulch	20.32	50.99	72.19
rock-bed	0	0	0
log	11.4	16.75	21.29
bicycle	nan	nan	nan
person	0.0	0.67	0.0
fence	19.12	28.56	32.51
bush	9.87	10.2	43.47
sign	0	0	0
rock	16.56	20.87	23.23
bridge	nan	nan	nan
concrete	0.0	0.63	1.82
picnic table	0	0	0

Tab. 11: Results per class of the Segformer model on RUGD dataset

A.14 Qualitative result of DAFormer RUGD to RUGD augmented with meta-classes

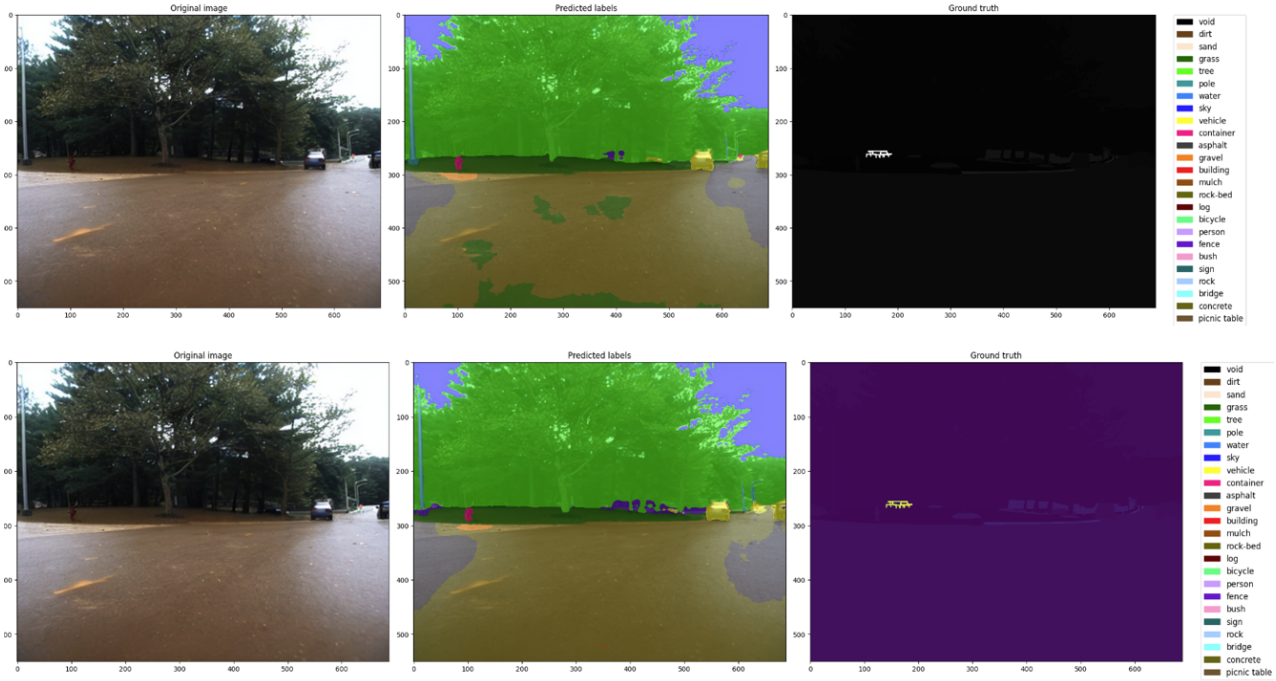

Fig. 47: The left-hand column represents the original image augmented of the environment park-8 flooded, the right-hand column the ground truth and the middle column the predicted segmentation mask. In this column, the DAFormer predictions are shown in the top row and the oracle predictions in the bottom row.

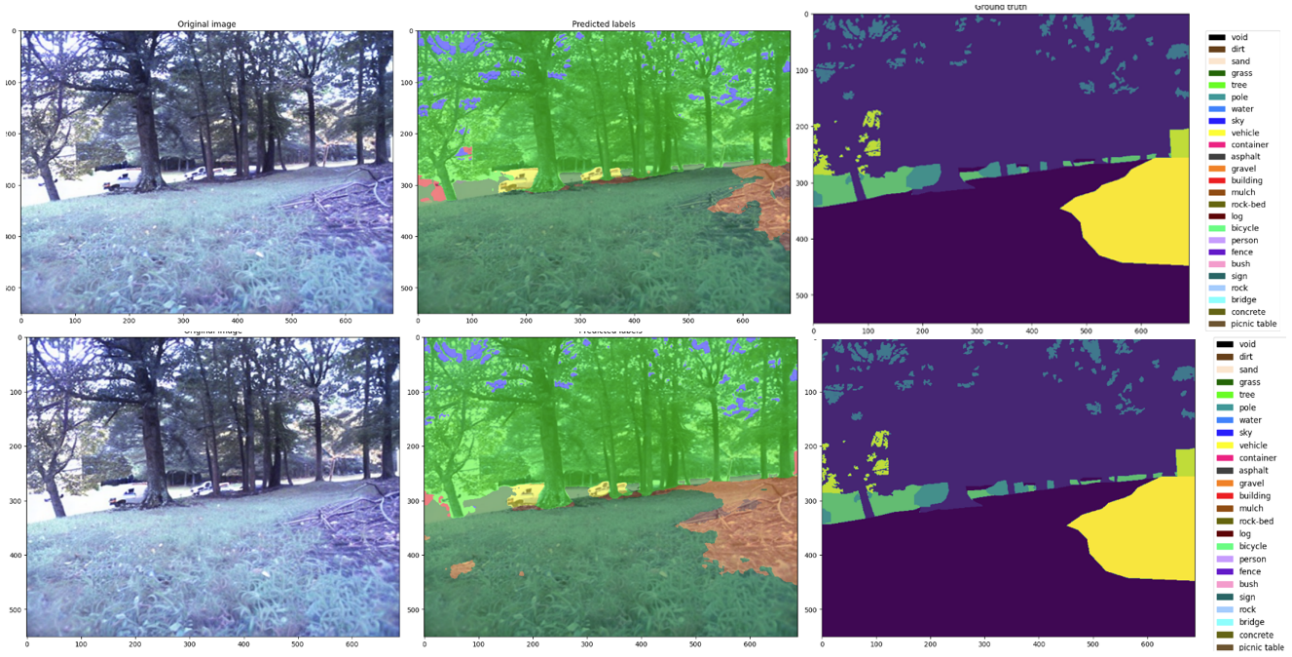

Fig. 48: The left-hand column represents the original image augmented of the environment park-8 winter, the right-hand column the ground truth and the middle column the predicted segmentation mask. In this column, the DAFormer predictions are shown in the top row and the oracle predictions in the bottom row.

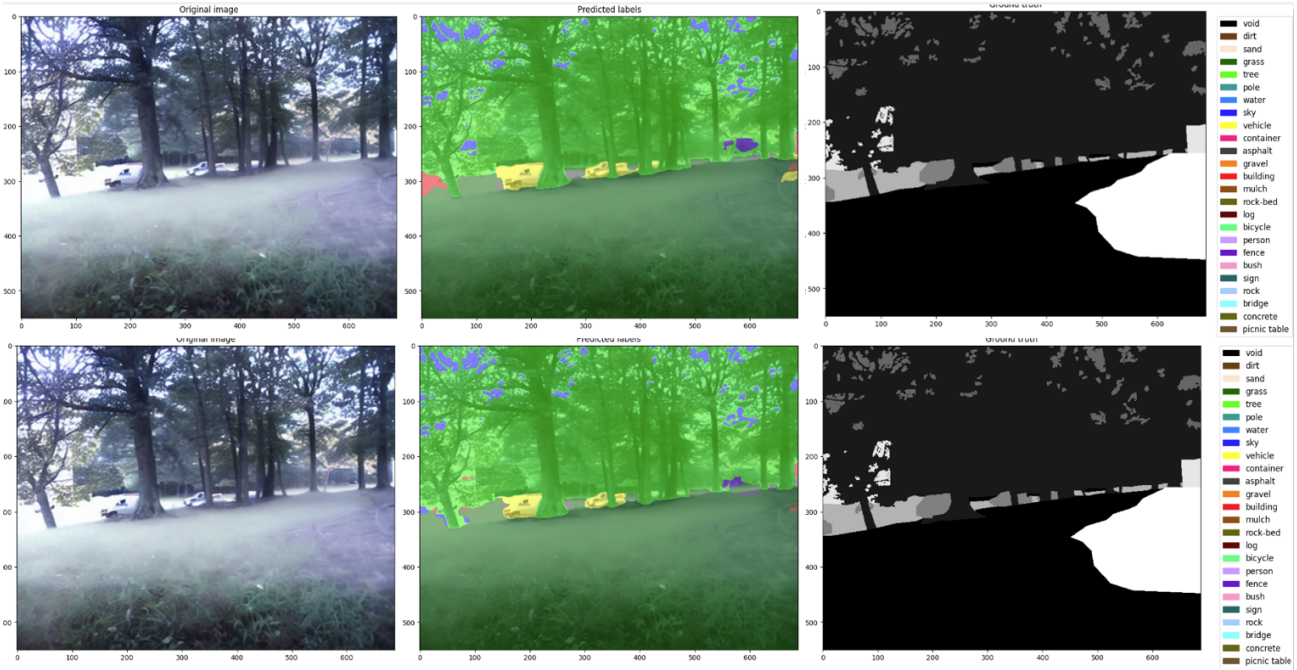

Fig. 49: The left-hand column represents the original image augmented of the environment park-8 fog, the right-hand column the ground truth and the middle column the predicted segmentation mask. In this column, the DAFormer predictions are shown in the top row and the oracle predictions in the bottom row.

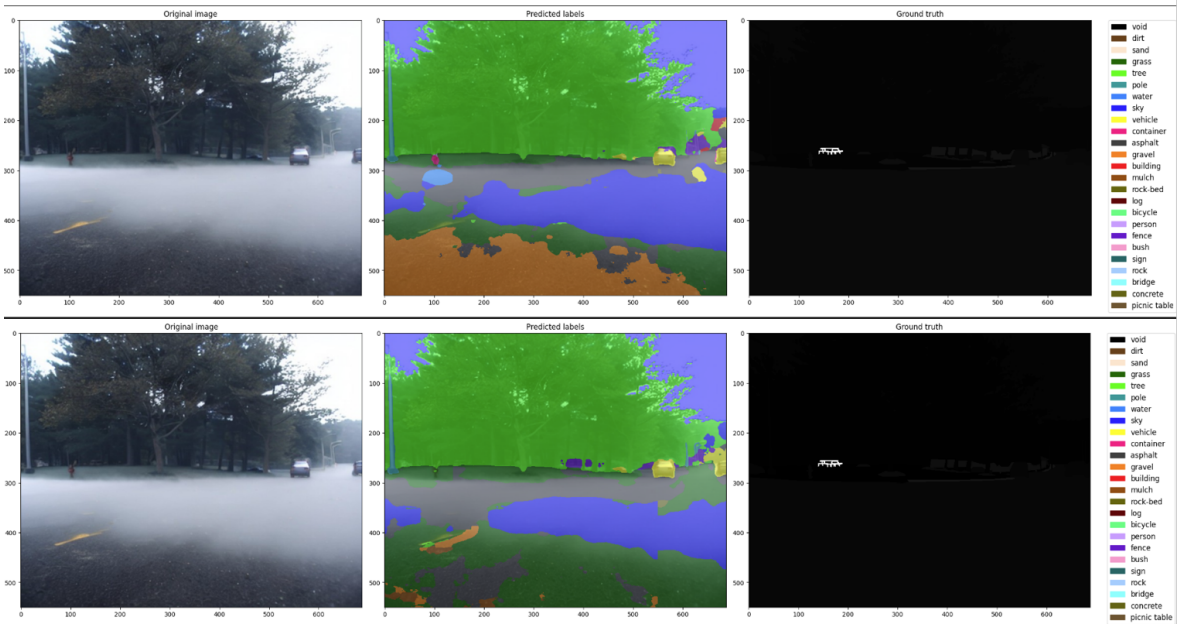

Fig. 50: The left-hand column represents the original image augmented of the environment park-8 fog, the right-hand column the ground truth and the middle column the predicted segmentation mask. In this column, the DAFormer predictions are shown in the top row and the oracle predictions in the bottom row.

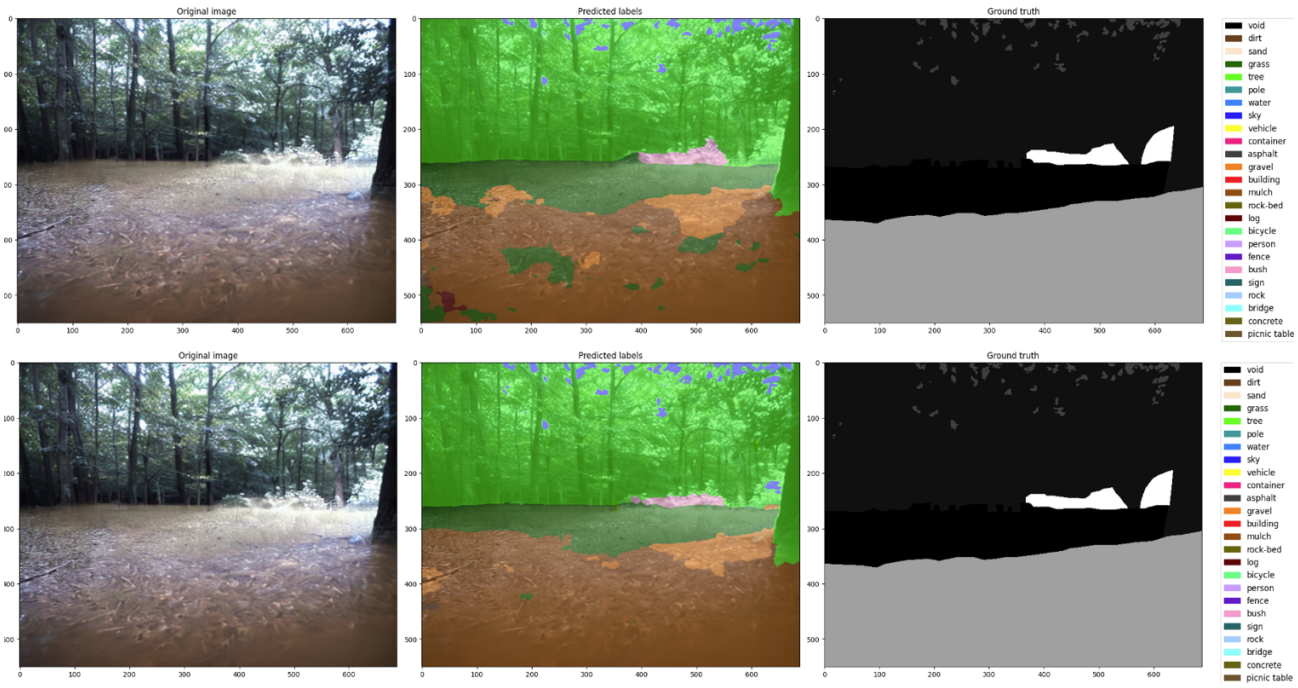

Fig. 51: The left-hand column represents the original image augmented of the environment trail-5 flooded, the right-hand column the ground truth and the middle column the predicted segmentation mask. In this column, the DAFormer predictions are shown in the top row and the oracle predictions in the bottom row.

A.15 Barometer DAFormer RUGD to Samba with IoU test results by class - Quantitative result

Class	Source only	DAFormer	Oracle
navigable	75.0	79.88	80.03
nonavigable	77.16	81.55	92.03

Tab. 12: Results per class of the Segformer model on RUGD dataset

A.16 Barometer DAFormer RUGD to RUGD augmented with IoU test results by class - Quantitative result

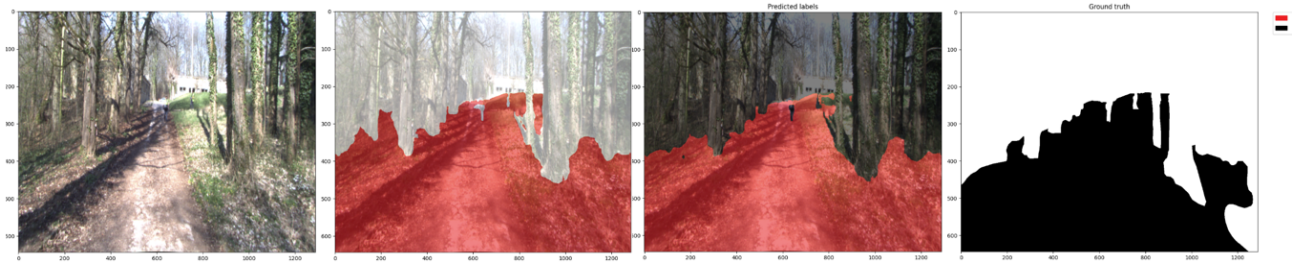

Fig. 52: First example - In the order of display of the images, there is the original image, the semantic prediction of the oracle, the semantic prediction of the DAFormer and finally lastly the ground truth. The red color corresponds to the navigable class

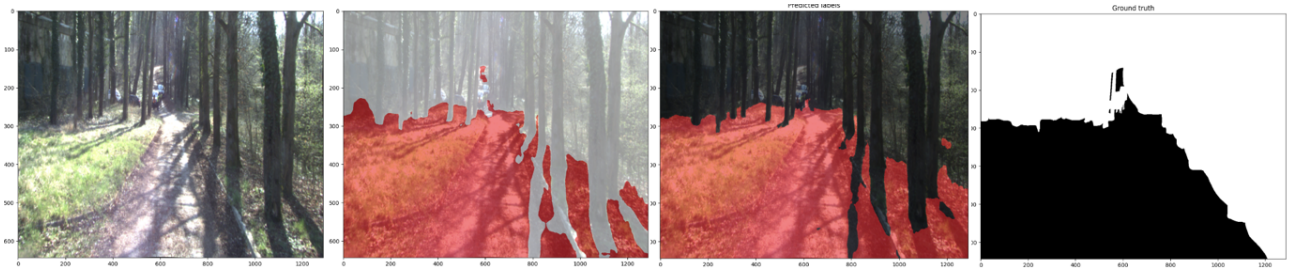

Fig. 53: Second example - In the order of display of the images, there is the original image, the semantic prediction of the oracle, the semantic prediction of the DAFormer and finally lastly the ground truth. The red color corresponds to the navigable class.

A.17 Review of the three principal UDA Barometers obtained

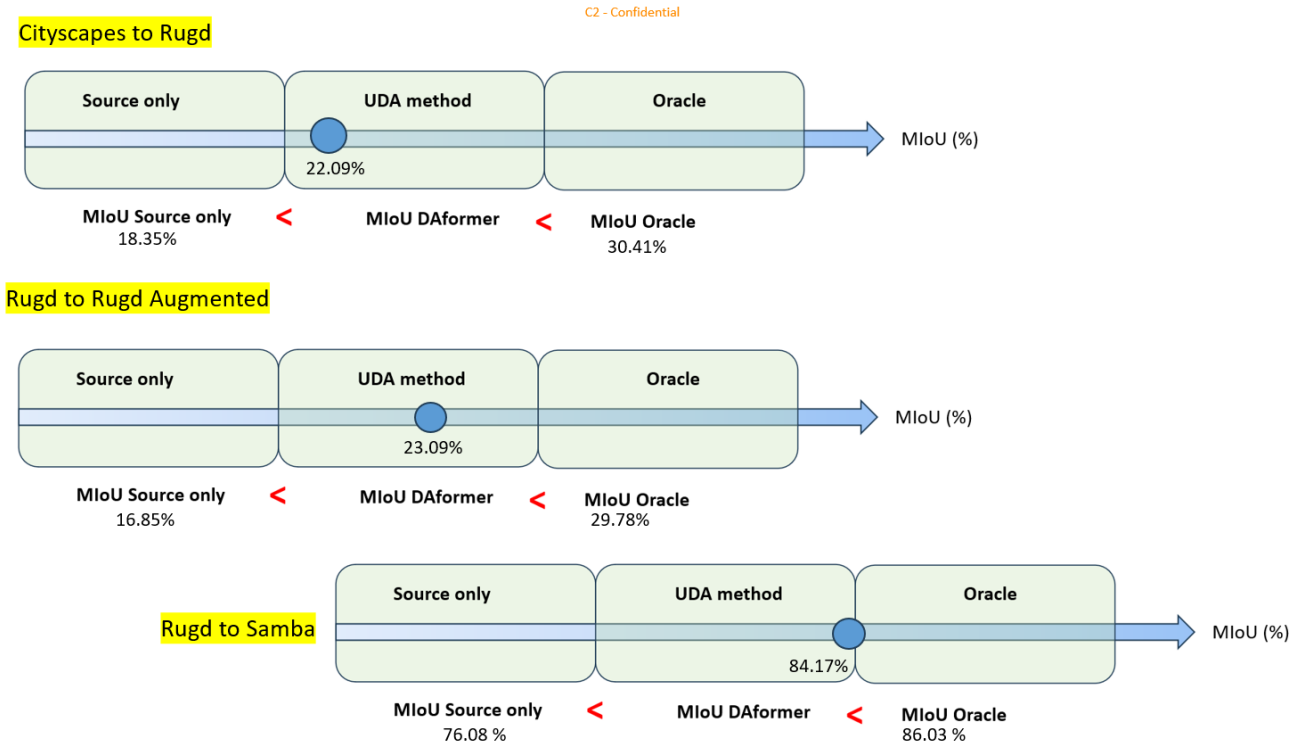

Fig. 54: Overview of the most important barometers obtained

References

[1] Joseph Gonzalez Sanjit A. Seshia Kurt Keutzer Sicheng Zhao, Bichen Wu. Unsupervised domain adaptation: from simulation engine to the real world. 2018.

[2] Comparaison Traditional ML vs Transfer Learning. https://mangastorytelling.tistory.com/entry/itfind-

[3] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. 2020.

[4] Ramesh Nallapati Andrew Arnold and William W. Cohen. A comparative study of methods for transductive transfer learning. 2007.

[5] Kuniaki Saito Neela Kaushik Judy Hoffman Kate Saenko Xingchao Peng, Ben Usman. Syn2real: A new benchmark forsynthetic-to-real visual domain adaptation. 2018.

[6] Categorization of Domain Adaptation methods. https://www.v7labs.com/blog/domain-adaptation-guide.

[7] Han Zhao Haoxiang Wang, Bo Li. Understanding gradual domain adaptation: Improved analysis, optimal path and beyond. 2022.

[8] Aidong Men Yang Liu Yinsong Xu, Zhuqing Jiang and Qingchao Chen. Delving into the continuous domain adaptation.

[9] Kate Saenko Trevor Darrell Eric Tzeng, Judy Hoffman. Adversarial discriminative domain adaptation. 2007.

[10] Lili Ju Hao Guo1 Song Wang Xinyi Wu1, Zhenyao Wu1. Dannet: A one-stage domain adaptation network for unsupervised nighttime semantic segmentation. 2007.

[11] Kui Jia Hui Tang. Discriminative adversarial domain adaptation. 2019.

[12] Zhixiang Wei Xin Jin Xiao Tan Yi Jin† Enhong Chen Lin Chen, Huaian Chen. Reusing the task-specific classifier as a discriminator: Discriminator-free adversarial domain adaptation. 2022.

[13] Choon Hui Teo Bernhard Scholkopf Arthur Gretton, Kenji Fukumizu. A kernel statistical test of independence. 2007.

[14] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. 2016.

[15] Ugur Halici Ibrahim Batuhan Akkaya1, Fazil Altinel. Self-training guided adversarial domain adaptation for thermal imagery. 2021.

[16] Ibrahim Batuhan Akkaya Fazil Altinel. Adversarial domain adaptation enhanced via self-training. 2021.

[17] Harri Valpola Antti Tarvainen. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. 2018.

[18] Luc Van Gool Lukas Hoyer, Dengxin Dai. Daformer: Improving network architectures and training strategies for domain-adaptive semantic segmentationn. 2022.

[19] Zhiding Yu Anima Anandkumar Jose M. Alvarez Ping Luo Enze Xie, Wenhai Wang. Segformer: Simple and efficient design for semantic segmentation with transformers. 2021.

[20] Alexander Kolesnikov Dirk Weissenborn Xiaohua Zhai Thomas Unterthiner Mostafa Dehghani Alexey Dosovitskiy, Lucas Beyer. An image is worth 16x16 words: Transformers for image recognition at scale. 2021.

[21] Improving Network Architectures Training for Semantic Segmentation Lukas Hoyer CVPR SDAS 2023. https://www.youtube.com/watch?v=pz2ntag0qaclist=plbq8cd-autbu4ndxinpuqnyh1bmxpqzrindex=3t=601s.

[22] Juliano Pinto1 Lennart Svensson Volvo Cars Wilhelm Tranheden, Viktor Olsson. Dacs: Domain adaptation via cross-domain mixed sampling. 2020.

[23] Tim Br¨odermann Luc Van Gool David Bruggemann, Christos Sakaridis. Contrastive model adaptation for cross-condition robustness in semantic segmentation. 2023.

[24] Francois Fleuret Prabhu Teja. Uncertainty reduction for model adaptation in semantic segmentation. 2021.

[25] Chang Liu Yidong Ouyang Tao Qin Jindong Wang, Cuiling Lan. Generalizing to unseen domains: A survey on domain generalization. 2022.

[26] Na Zhao Nicu Sebe Yuyang Zhao, Zhun Zhong and Gim Hee Lee. Style-hallucinated dual consistency learning for domain generalized semantic segmentation. 2022.

[27] Serge Belongie Xun Huang. Arbitrary style transfer in real-time with adaptive instance normalization. 2017.

[28] Dengxin Dai Lukas Hoyer and Luc Van Gool. Domain adaptive and generalizable network architectures and training strategies for semantic image segmentation. 2023.

[29] Haoran Wang Luc Van Gool Lukas Hoyer, Dengxin Dai. Mic: Masked image consistency for context-enhanced domain adaptation. 2023.

[30] Prune Truong Luc Van Gool David Bruggemann, Christos Sakaridis. Refign: Align and refine for adaptation of semantic segmentation to adverse conditions. 2023.

[31] Dengxin Dai Lukas Hoyer and Luc Van Gool. Hrda: Context-aware high-resolution domain-adaptive semantic segmentation. 2022.

[32] mmseg. https://mmsegmentation.readthedocs.io/en/latest/.

[33] Christian Szegedy Sergey Ioffe. Batch normalization: Accelerating deep network training by reducing internal covariate shift.

[34] Zhiding Yu3 Anima Anandkumar3 4 Jose M. Alvarez3 Ping Luo Enze Xie1, Wenhai Wang2. Segformer: Simple and efficient design for semantic segmentation with transformers.

[35] Sebastian Ramos Timo Rehfeld Markus Enzweiler Rodrigo Benenson Uwe Franke Stefan Roth Bernt Schiele Marius Cordts, Mohamed Omran. The cityscapes dataset for semantic urban scene understanding. 2016.

[36] Official site of the ACDC dataset. https://acdc.vision.ee.ethz.ch/.

[37] John G. Rogers III David Han Maggie Wigness, Sungmin Eum and Heesung Kwon. A rugd dataset for autonomous navigation and visual perception in unstructured outdoor environments. 2019.

[38] Jong Chul Ye Gihyun Kwon. Clipstyler: Image style transfer with a single text condition. 2021.

[39] Andrei Bursuc Patrick Pérez Mohammad Fahes, Tuan-Hung Vu. PØda: Prompt-driven zero-shot domain adaptation. 2023.

[40] Varun Jampani Yael Pritch Michael Rubinstein Kfir Aberman Nataniel Ruiz, Yuanzhen Li. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. 2022.

[41] Alexei A. Efros NTim Brooks, Aleksander Holynski. Instructpix2pix: Learning to follow image editing instructions. 2022.

[42] Hugging Face. https://huggingface.co/spaces/timbrooks/instruct-pix2pix. 2022.

[43] Tinghui Zhou Alexei A. Efros Phillip Isola, Jun-Yan Zhu. Image-to-image translation with conditional adversarial networks. 2018.

[44] Ilya Loshchilov Frank Hutter. Decoupled weight decay regularization. 2018.

[45] Yarin Gal A. Tuan Nguyen, Toan Tran. Kl guided domain adaptation. 2022.

[46] Tuan Nguyen Huy Nguyen Hung Bui Nhat Ho Dinh Phung Trung Le, Dat Do. On label shift in domain adaptation via wasserstein distance. 2022.

[47] Nikhila Ravi Hanzi Mao Chloe Rolland Laura Gustafson Tete Xiao Spencer Whitehead Alexander C. Berg Wan-Yen Lo Piotr Doll Ross Girshick Alexander Kirillov, Eric Mintun. Segment anything. 2023.