

Alma Mater Studiorum Università di Bologna  
Archivio istituzionale della ricerca

Domain Adaptation for Image Classification of Defects in Semiconductor Manufacturing

This is the final peer-reviewed author's accepted manuscript (postprint) of the following publication:

*Published Version:*

Poniatowski, A., Gentner, N., Barusco, M., Pezze, D.D., Salti, S., Susto, G.A. (2025). Domain Adaptation for Image Classification of Defects in Semiconductor Manufacturing. IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, 23, 3818-3828 [10.1109/tase.2025.3621854].

*Availability:*

This version is available at: <https://hdl.handle.net/11585/1048829> since: 2026-02-25

*Published:*

DOI: <http://doi.org/10.1109/tase.2025.3621854>

*Terms of use:*

Some rights reserved. The terms and conditions for the reuse of this version of the manuscript are specified in the publishing policy. For all terms of use and more information see the publisher's website.

This item was downloaded from IRIS Università di Bologna (<https://cris.unibo.it/>).  
When citing, please refer to the published version.

(Article begins on next page)

# Domain Adaptation for Image Classification of Defects in Semiconductor Manufacturing

Adrian Poniatowski, Natalie Gentner, Manuel Barusco, Davide Dalle Pezze, Samuele Salti and Gian Antonio Susto *Senior Member, IEEE*

**Abstract**—In the semiconductor sector, due to high demand but also strong and increasing competition, time to market and quality are key factors in securing significant market share in various application areas. Thanks to the success of deep learning methods in recent years in the computer vision domain, Industry 4.0 and 5.0 applications, such as defect classification, have achieved remarkable success. In particular, Domain Adaptation (DA) has proven highly effective since it focuses on using the knowledge learned on a (source) domain to adapt and perform effectively on a different but related (target) domain. By improving robustness and scalability, DA minimizes the need for extensive manual re-labeling or retraining of models. This not only reduces computational and resource costs but also allows human experts to focus on high-value tasks. Therefore, we tested the efficacy of DA techniques in semi-supervised and unsupervised settings within the context of the semiconductor field. Moreover, we propose the DBACS approach, a CycleGAN-inspired model enhanced with additional loss terms to improve performance. All the approaches are studied and validated on real-world Electron Microscope images, considering the unsupervised and semi-supervised settings, proving the usefulness of our method in advancing DA techniques for the semiconductor field.

**Note to Practitioners**—Computer Vision-based inspection systems are fundamental to ensuring quality in many productive systems. Reliable image labeling is unfortunately very hard to automatize in complex manufacturing scenarios and often still relies on human experts, especially for complex and varying image content. Hence, limited or absent labels pose a significant challenge. In this work, a Deep Learning-based approach is proposed for image classification that can exploit heterogeneous data. In the context of fault detection, the proposed approach allows for enhanced reliability and stability in the classification, which is of crucial importance for dependable automated decision-making in manufacturing. The approach has been validated on real-world data associated with scanning electron microscope images collected in a productive semiconductor manufacturing environment.

**Index Terms**—Computer Vision, Convolutional Neural Network, Deep Learning, Defect Classification, Domain Adaptation, Industry 4.0, Pseudo Labeling, Scanning Electron Microscope Images

## I. INTRODUCTION

SEMICONDUCTORS are constructed by linking circuit structures on various layers built upon a thin slice of semiconductor material called a wafer. The creation of those layers,

First, second, third, and fourth authors equally contributed to the present work.

A. Poniatowski and N. Gentner are with Infineon Technologies AG, Germany.

N. Gentner, M. Barusco, D. Dalle Pezze and G.A. Susto are with the Department of Information Engineering, University of Padova, Italy.

A. Poniatowski and S. Salti are with the Department of Computer Science and Engineering, University of Bologna, Italy.

including their complex individual structures and patterns, is done in a sequential manner through a combination of the steps. The repetition of process steps and re-entry into the production flow is part of the advanced and highly specialized fabrication process.

Metrology is a set of quality assurance steps carried out in production to control and calibrate machines, process settings, process outcomes, and wafer properties. Within this framework, Defect Detection and Classification is a fundamental part of metrology to ensure high product quality and production performance. In this context, the use of Scanning Electron Microscope (SEM) images of the wafer can be used (see Figure 1 and 3). The collected data is examined and divided into different defect types before corrective actions or the declaration of 'defective' products are triggered (the entire inspection procedure is depicted in Figure 1).

Industry 4.0, which is enabled by technologies such as Deep Learning, enabled automatic defect classification, which has become cost-effective and widely adopted [1]. However, a series of challenges hinder the scalability of these models, such as the differences in data distributions combined with the abundance of data not labeled [2].

Domain Adaptation (DA) is extremely fitted for Industry 5.0 applications which focus on human-centric approaches, sustainability, and resilience. Indeed, DA focuses on using the knowledge learned on a (source) domain to adapt and perform effectively on a different, but related (target) domain. DA minimizes the need for extensive manual re-labeling or re-training of models, allowing human experts to focus on high-value tasks and improving robustness and scalability.

Specifically, in the context of the semiconductor field, to enable fab-wide usage of highly automated classification models, deployed methods need to be transferable and applicable when confronted with new optical properties and different image structures generated from technological diversity, as well as recurring production steps in different stages of the manufacturing process.

Therefore, in this work, we propose to study the effectiveness of DA techniques in the context of the semiconductor domain. In particular, we focus on state-of-the-art models able to perform in both the semi-supervised domain adaptation (SSDA) and the unsupervised domain adaptation (UDA) settings. We focus on the *AdaMatch* [3] approach for the pseudo-labeling group and *DBACS* [4] as a generative and adversarial model. While both have been proven to be of extreme applicability, they have never been applied so far in the context of image classification problems in semiconductor

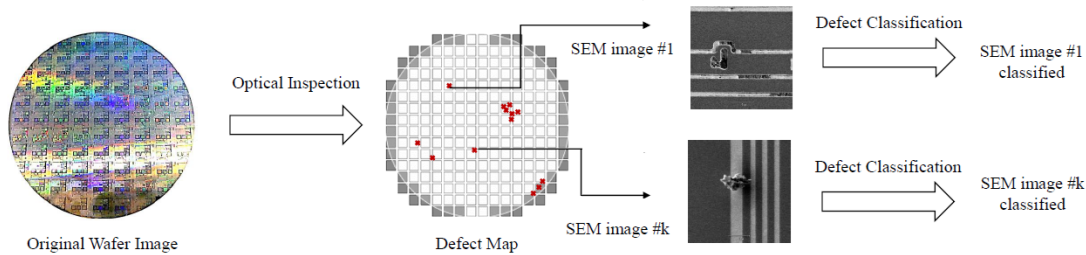


Fig. 1: Defect classification procedure for the semiconductor manufacturing process at hand: (i) an optical inspection tool locates defects (marked in red) by comparing multiple neighboring areas on a wafer and looking for anomalies; (ii) high-resolution SEM images are taken at each defect location. (iii) defects are classified: differences in the image background structure are visible, typically because images are taken in different positions on the wafer, or more generally, different wafers belong to different technologies, hence showing variations in their surface structure.

manufacturing.

In particular, in this work, we consider an evolution of the DANN approach [5] called *DANN-based Alignment with Cyclic Supervision* (DBACS)[4]. This method was only considered for time-series data, and it is studied for the first time in the image domain. While this is a CycleGAN-inspired model, we enhance and adapt the original version for computer vision by adding the following modifications: cyclic adaptation, unpaired sample mapping and feature matching loss, identity loss, and multi-scale structure similarity loss. These modifications improved the final performance.

The main contributions of this work are summarized as follows:

- We test and study for the first time the use of DA techniques to solve the defect classification problem in the semiconductor manufacturing field for the SEM images.
- We introduce the use of the DBACS approach in the Computer Vision domain by adapting its architecture and by considering additional loss terms.
- The effectiveness of our approach is validated on real-world Electron Microscope images, considering the unsupervised and semi-supervised settings, proving also the usability and usefulness of DA techniques for the SEM images.

Furthermore, we have made all the code used in our study, including the DBCAS implementation, publicly accessible<sup>1</sup>. This allows researchers and practitioners to utilize our implementation for further advancements.

The rest of the paper is organized as follows: In Section II, we review the literature concerning the semiconductor sector and the domain adaptation field. Section IV defines the problem formally and presents our approach. Then Section V describes the methods used as a comparison. Section VI provides detailed information about the data used, the experimental setup, including the considered models, and the specific hyperparameters employed for each DA technique. Section VII shows and discusses the obtained results. Finally, Section VIII summarizes our findings, and future research directions are presented.

## II. RELATED WORK

We delineate the relevant literature as follows. In Section II-A, an overview of previous works to solve the image classification problem in semiconductor manufacturing is provided. Then we discuss Domain Adaptation with a focus on the two main families, Pseudo-labeling in Section II-B1 and Generative Adversarial Models in Section II-B2

Task	Labeled	Unlabeled	Distributions
SSL	source	target	source = target
UDA	source	target	source $\neq$ target
SSDA	source + target	target	source $\neq$ target

TABLE I: Comparison of different learning paradigm: Self-Supervised Learning (SSL), Unsupervised Domain Adaptation (UDA), Semi-Supervised Domain Adaptation (SSDA).

### A. Image Classification in Semiconductor Manufacturing

Image classification in semiconductor manufacturing is mainly applied for two different tasks: *image defect classification* and *wafer-level/wafer map defect pattern classification*. In image defect classification, images are taken at relevant spots in the wafer and then classified into a priori known defects (see SEM image in Figure 1). In wafer-level/wafer map defect pattern classification, it is considered a map of the wafer where pre-identified anomalies/potential defects have already been identified, and the task is to classify such maps w.r.t. known defect patterns (see defect map in Figure 1).

Some research focusing on wafer-level/wafer map defect pattern classification was published in recent years, eventually also driven by the wafer map dataset WM-811k made publicly available by [6].

Deep Learning exhibits considerable effectiveness in numerous Industry 4.0 and 5.0 applications, addressing a wide range of problems. For instance, in predictive maintenance, Deep Learning has become a preferred choice due to its high performance and automatic feature extraction capabilities [7]. Similarly, in image defect classification, Deep Learning approaches help to drastically reduce the amount of time spent by human operators in manually tagging images [8]. It also shows that CNN-based transfer learning methods can classify

<sup>1</sup>[https://bitbucket.org/papers\\_vad\\_group/dbacs/src/main/](https://bitbucket.org/papers_vad_group/dbacs/src/main/)

microscopic defect images with high accuracy.

In [9] authors proposed a stacked hybrid CNN that exploits an attention mechanism. Also, in this case, CNNs and Deep Learning approaches are the typical choices [10], [11] for performing the classification task. [10] additionally demonstrates that a single CNN model can extract effective features for defect classification without using additional feature extraction algorithms. An example of the application of pseudo-labeling is [12]. However, only a few works apply really deep architectures in this context. In this work, we consider the application of well-known DL models such as MobileNet, ResNet50, and Resnet101.

## B. Domain Adaptation

Typically in DL, higher accuracy can be achieved when more data, but also unlabeled data is available and used. When the unlabeled data is assumed to come from the same distribution as the labeled data, we talk about Self-Supervised Learning (SSL). When, instead, it is assumed that the unlabeled data comes from a different distribution, then we are in the UDA setting, where the notion of *source* and *target domains* is used. In real-world applications, the setting called SSDA usually occurs, where the source and the target distributions differ, and there is a portion, usually small, of the target domain data that is labeled. A taxonomy of all the settings is reported in Table I.

1) *Pseudo-labeling*: In general, Pseudo-labeling (PL) is a well-known and often employed technique for semi-supervised and self-supervised learning. Pseudo-labeling [13] is first applied to SSL (learning paradigm, see Table I) and consists of producing artificial labels for unlabeled images and in training the model to predict the artificial labels when fed unlabeled images as input. It uses the model class prediction as a label to train against if it is above a certain probability threshold. Several methods have been proposed in the literature that exploit this concept.

For example, *AugMix* improves model robustness and uncertainty estimation by combining data modification in more detailed image augmentation with Jensen-Shannon divergence as consistency loss. *FixMatch* [14] is a successful attempt to address and simplify SSL. It combines various semi-supervised learning techniques from [15] and [16]. Similarly to *ReMixMatch* [17], it also leverages *CTAugment* and especially *Cutout* [18] to produce heavily distorted, *strongly augmented*, versions of an image and compare them to *weakly augmented* ones. *AdaMatch* [3] extends *FixMatch* by also taking into consideration the domain shift between the labeled and the unlabeled data, making it a suitable model for UDA (see Table I). Since this is the current state-of-the-art for semi-supervised and unsupervised domain adaptation for vision classification tasks, it is selected as one of our applied models. Indeed, since the considered method (DBACS) can perform SSDA and UDA, this is the best state-of-the-art model for this category to compare with. A detailed description of *AdaMatch* is given in Section V. *AdaEmbed*, [19] tackles Semi-Supervised Domain Adaptation (SSDA) and improves performance by learning a shared embedding space and generating accurate pseudo-labels.

2) *Generative and Adversarial Models*: When considering image-to-image (I2I) translation, *pix2pix* [20] or conditional GAN (cGAN) [21], the Generator component of this GAN-based [22] models is an autoencoder-like network that maps an input image from a known source distribution to another target distribution. Such image translations are used for changing properties of single objects or the style of images when supervised learning is done, or *paired* images from source and target are available. In the case of unpaired I2I no such association between the input/source and output/target images is available. *CycleGAN* [23] tackles this problem by introducing a second generator-discriminator couple and trying to learn a bijective mapping by translating the output of the original generator back to the original input space. A *cycle-consistency* loss is introduced to compare an image from the source or target domains with the same image after forward and backward translation. The concepts of I2I tasks, cGANs, and *CycleGANs* can be used in the Domain Adaptation field, where source and target domains are different, and it is possible to transform the samples from one domain to another. This allows a model, trained on the source domain samples, to perform well on the target domain samples thanks to this transformation.

Based on domain adaptation theory, Domain Adversarial Neural Networks (DANN) [5] are introduced. They target the model generalization issue when facing data differences originally within the training and test data. In their settings, an adversarial training approach similar to GANs is introduced. The DANN approach used to tackle the UDA problem for image classification focuses on the alignment between the source and target domains of the latent features extracted by the convolutional part of a deep neural network classifier, called the feature extractor. This alignment can be obtained by measuring and thus improving the discrepancy between the latent feature distributions of the source and target domains. Different probability distance measures are adopted: [24] uses the maximum mean discrepancy, [25] uses correlation alignment between domains, and [26] measures the discrepancy by adopting multiple task classifiers. *AD-Aligning* [27] is another adversarial approach that emulates human-like generalization by combining adversarial training with source-target domain alignment. By pretraining with *Coral* loss and standard loss, *AD-Aligning* aligns target domain statistics with those of the pretrained encoder, preserving robustness while accommodating domain shifts. *AVATAR* [28] instead introduces a domain-adaptive approach combining adversarial learning, self-supervised strategies, and sample selection for robust performance, especially in unsupervised domain adaptation scenarios. *Dann-based Alignment Model (DBAM)* [29] is an adversarial domain adaptation model inspired by DANN and semiconductor manufacturing modelling problems. Originally, it is applied to time series data and used to solve a *Virtual Metrology/Soft Sensing* task, to improve the manufacturing process monitoring and its related operations by allowing interpretability and comparability of input and aligned output. Since it aligns the original input spaces, it is similar to I2I-based GANs with an autoencoder-like generator, and the aligned features can still be compared and are physically interpretable.

In this paper, an extended version called DANN-based Alignment with Cyclic Supervision (DBACS) [4] introduced for heterogeneous domain adaptation on time series data, is enhanced to be used for an unpaired image-to-image translation task. The model tackles the domain adaptation problem by transforming the source domain images into the target domain images. This allows a classifier trained on the source domain to correctly classify defects on the target domain since the input images are transformed to the source domain. To the best of our knowledge, this is the first application of a domain adaptation method on defect classification in semiconductor manufacturing. A detailed description of the method can be found in Section V

### III. OTHER APPROACHES

Recent advancements in Unsupervised Domain Adaptation have focused on avoiding the need for source data on training, with methods such as ProxyMix [30], which integrate a mixup regularization to enhance domain adaptation, improving model robustness against noisy pseudo labels and domain discrepancies. [31] improves UDA by addressing the issues of inconsistent learning objectives and noisy pseudo labels. It enhances domain adaptation by simultaneously learning instance discrimination and semantic alignment across domains while removing false positive and negative pairs through topology-based selection. Instead, in the field of Semi-Supervised domain adaptation, [32] introduces a cross-domain feature alignment strategy that incorporates label information into model adaptation. This approach enables better cross-domain feature alignment and reduces label mismatch. Furthermore, Neighborhood Expansion leverages high-confidence pseudo-labeled samples from the target domain, enriching the label diversity and consequently boosting the performance of the adaptation model. Recently, the fields of Continual Learning and Domain Adaptation are overlapping, and [33] tackles domain adaptation with limited labeled target data by using a Conditional Variational Auto-Encoder (CVAE) and soft domain attention.

### IV. OUR APPROACH

In the following part, a mathematical formalization of the addressed problem is given in Section IV-A. In Section IV-B, we discuss foundational approaches, including the DANN-based Alignment Model (DBAM) and its extension, the DBACS method, highlighting their principles and architectures. Finally, in Section IV-C, we detail our contributions to DBACS by integrating novel modifications, such as cyclic adaptation, unpaired sample mapping, and additional loss functions.

#### A. Problem Formalization

In this work, the focus is to learn a model in a classification setting and to scale it from one labeled source domain to a target domain with a different data distribution, where the data are partially labeled (SSDA) or completely unlabeled (UDA). Let's define  $X$  as the input space,  $Y$  as the output space, and a *domain* as a distribution over  $X \times Y$ . A learning algorithm is

now provided with a data set  $S$  drawn i.i.d. from a domain  $D_S$  with  $X_S \times Y_S$ ,  $X_S \subset X$ ,  $Y_S \subset Y$ . For better differentiation,  $D_S$  is called *source domain*. In the SSL setting, we distinguish between labeled and unlabeled data and define  $S := SL \cup SU$  where  $SU$  stands for the unlabeled sample subset and  $SL$  for the labeled one. Without loss of generality for UDA and SSL,  $SU = \emptyset$  since the source domain is assumed to be labeled. Hence:

$$S = \{X_S, Y_S\} = \{X_{SL}, Y_{SL}\} = \{x_S^i, y_S^i\}_{i=1}^n \sim D_S, \quad (1)$$

with  $n$  being the number of drawn samples (all labeled) and therefore  $X_S = X_{SL} \subset X$ ,  $Y_S = Y_{SL} \subset Y$ . We assume to have two data sets. Let  $T = TL \cup TU$  be the second data set  $T$  drawn i.i.d. from a domain  $D_T$  called *target domain* with a distribution over  $X_T \times Y_T$ ,  $X_T \subset X$ ,  $Y_T \subset Y$ , and consisting of unlabeled  $TU$  and/or labeled  $TL$  samples.

$$TL = \{X_{TL}, Y_{TL}\} = \{x_T^j, y_T^j\}_{j=1}^{m-l} \sim D_T; \quad (2)$$

$$TU = \{X_{TU}\} = \{x_T^j\}_{j=m-l+1}^m \sim D_T^X; \quad (3)$$

with  $m$  being the number of drawn target samples and  $l$  the number of labeled samples, therefore  $X_{TL} \subset X_T \subset X$ ,  $Y_{TL} \subset Y_T \subset Y$  and  $X_{TU} \subset X_T \subset X$ . Let  $D_S^X$ ,  $D_S^Y$  and  $D_T^X$ ,  $D_T^Y$  be the marginal distributions of  $D_S$  and  $D_T$  over  $X_S$ ,  $Y_S$  and  $X_T$ ,  $Y_T$  respectively.

#### B. DBAM and DBACS methods

*Domain Adversarial Neural Networks* (DANN) [5] measure the domain discrepancy using a domain discriminator network combined with a suitable probability distance metric and by employing adversarial training. DANN-based Alignment Model (**DBAM**) is based on the DANN approach and it is presented in [29] as a GAN model. The idea is to keep the necessary information to allow interpretability and comparison of source and target while maintaining the high accuracy of a dedicated model trained on the source domain. It is originally applied on time series and stationary data: DBAM consists of 3 parts: a classifier/predictor  $f_{DB}$  that solves the task, a generator - also called aligner  $F$  - that maps the target domain to the source domain to enable classifier usage for mapped target data, and a discriminator (called discriminator A later) that distinguishes between source data and aligned target data. DBAM uses I2I-like translation, meaning it works with the original feature spaces for both source and target. In more detail, DBAM is mapping source input data into the original target feature space via an autoencoder-like generator network. Adversarial training is applied using the discriminator.

CycleGAN [23], a model designed for unpaired image-to-image translation by introducing cycle-consistency loss to map between two domains, is adopted and enhanced in this work. In particular, we exploit an extended version of DBAM called DANN-based Alignment with Cyclic Supervision (**DBACS**) [4]. Since the task asks for unpaired mapping, a second generator (called aligner  $G$ ) is used that translates the source to the target domain and the output of aligner  $F$  back to the target domain. Adversarial training in this direction is enabled by adding a second domain discriminator (called discriminator  $B$ ). By combining both aligners, it is possible to introduce

*cycle-consistency* by comparing source samples with their cycled samples and target samples with their cycled samples. A visualization of DBACS is presented in Figure 2.

### C. Our Approach

We extend the DBACS approach by using it for the computer vision domain. In particular, we introduced the following series of modifications: cyclic adaptation, unpaired sample mapping, identity loss, feature matching, and multi-scale structure similarity. We define the different parts of the final loss function  $\mathcal{L}_{final}$  used for training. We apply the same training routine as for DBAM [29] and DBACS [4], by using adversarial training, i.e., alternatively training discriminator and aligner parts with contradictory goals and a fixed ratio  $r_{adv}$ . The discriminator's objective is to maximize the distance between the distributions of the source and the aligned target. The aligner's objective is to minimize the distance between the source and aligned target distribution and to minimize the prediction error of the classifier model on the aligned target data. Let  $f_{cc} : X \rightarrow Y$  be the source data trained classification model with softmax output. Let  $F : X_S \rightarrow X_T$  define a statistical model from source domain to target domain,  $G : X_T \rightarrow X_S$  a statistical model from target to source domain,  $D_A : X_S \rightarrow \{0, 1\}$ ,  $D_B : X_T \rightarrow \{0, 1\}$  statistical model classifying source versus aligned target and target versus aligned source.

1) *DBACS - Classifier Loss*: Only applicable in case of partly labeled target data when semi-supervised learning is possible. The classifier part of  $\mathcal{L}_{final}$  is defined as:

$$\begin{aligned} \mathcal{L}_{cc} &= \mathcal{L}(f_{cc}, Y_{TL}) \\ &= \sum_{(x_{TL}, y_{TL}) \in T} (f_{cc}(F(x_{TL})), y_{TL}) \end{aligned} \quad (4)$$

where  $\mathcal{L}$  is the categorical cross entropy loss. When used, it supports task-specific alignment of target data.

2) *DBACS - Adversarial Loss*: The adversarial loss is applied to the output of the Discriminator. During Generator training, it is minimized, during Discriminator training it is maximized (or its negative value minimized). First, we define it for the target to source alignment where F tries to map the target domain to the source domain, then for source to target alignment with G:

$$\mathcal{L}_{adv_S}(F, D_A, X_S, X_T) = \log(1 - D_A(F(x_T))) + \log(D_A(x_S)) \quad (5)$$

$$\mathcal{L}_{adv_T}(G, D_B, X_S, X_T) = \log(1 - D_B(G(x_S))) + \log(D_B(x_T)) \quad (6)$$

This is the original loss function used in GANs [22] and in DANN [5] for classification.

3) *DBACS - Cycle Consistency Loss*: The cycle-consistency loss minimizes two differences: the one between source data  $x_S \in X_S$  and their cycled version  $F(G(x_S))$  and the one between target samples  $x_T \in X_T$  and their cycled version  $G(F(x_T))$ :

$$\mathcal{L}_{cyc}(G, F) = \mathcal{L}_1(F(G(x_S)), x_S) + \mathcal{L}_1(G(F(x_T)), x_T) \quad (7)$$

with  $\mathcal{L}_1$  is chosen as L1-norm (see [23]).

4) *DBACS - Identity Loss*: Also used in [23] in case of distribution overlap between source and target domain. Minimizes the differences between  $x_T$  and  $G(x_T)$  and between  $x_S$  and  $F(x_S)$  when fed with  $x_T \in X_S$  and  $x_S \in X_T$ .

$$\mathcal{L}_{id}(G, F) = \mathcal{L}_1(F(x_S), x_S) + \mathcal{L}_1(G(x_T), x_T). \quad (8)$$

5) *DBACS - Additional Loss*: Two additional losses inspired by [34] are added. In case the mapping between the two domains is not bijective, the cyclic loss should aim to preserve the most important information by better preserving features visible to humans rather than noisy, high-frequency information. Hence *Multi Scale Structure Similarity loss* (MS-SSIM) is added in addition to the cyclic loss.

$$\begin{aligned} \mathcal{L}_{ssim}(G, F) &= (1 - \mathcal{L}_{ssim}(F(G(x_S)), x_S)) \\ &\quad + (1 - \mathcal{L}_{ssim}(G(F(x_T)), x_T)). \end{aligned} \quad (9)$$

*Feature Matching loss* is introduced in order to encourage aligned and original samples to produce similar activation at each layer of the discriminator instead of just output layer.

$$\mathcal{L}_{fm_A}(F, D_A) = \frac{1}{l-1} \sum_{l=1}^{n-1} \|a_l(x_S) - a_l(F(x_T))\|_2^2 \quad (10)$$

$$\mathcal{L}_{fm_B}(G, D_B) = \frac{1}{l-1} \sum_{l=1}^{n-1} \|a_l(x_T) - a_l(G(x_S))\|_2^2 \quad (11)$$

where  $a_l \in D_A$  represents the raw activation of the  $l^{th}$  layer of the discriminator  $D_A$ , and  $l$  is the total number of discriminator's layers, including output.

The final loss  $\mathcal{L}_{final}$  is defined as:

$$\begin{aligned} \mathcal{L}_{final} &= \lambda_{cc} \mathcal{L}_{cc} + \lambda_{adv} \mathcal{L}_{adv} + \lambda_{cyc} (\mathcal{L}_{cyc} + \mathcal{L}_{ssim}) \\ &\quad + \lambda_{id} \mathcal{L}_{id} + \lambda_{fm} \mathcal{L}_{fm} \end{aligned} \quad (12)$$

with

$$\mathcal{L}_{adv} = \mathcal{L}_{adv_S} + \mathcal{L}_{adv_T}$$

$$\mathcal{L}_{fm} = \mathcal{L}_{fm_A} + \mathcal{L}_{fm_B}$$

and where  $\lambda_{(\cdot)}$  represents the weight assigned to each corresponding loss term. We also follow some general architecture recommendations: both aligners have a U-Net [35] like architecture taken from [20] but without cropping so that the skip connection could be applied. For both discriminators the architecture is taken from [34], which improves CycleGAN's shape deformation by introducing *dilated convolutions*, often used in semantic segmentation: this allows a bigger receptive field with respect to the input image at the same cost of a convolution but with a much smaller filter.

## V. COMPARED APPROACHES

We discuss the first group of DA techniques called Pseudo Labeling in Section V-A. Then we dedicated a detailed description of the state-of-the-art approach, AdaMatch in Section V-B.

**Algorithm 1** DBACS for defect classification

- 
- 1: Prepare source trained classifier  $f_{cc}$  with frozen weights; two discriminators  $D_A, D_B$ ; two aligners  $F, G$ ; assign loss weights, learning rate, optimizer
  - 2: **for**  $iteration = 1, 2, \dots, epochs$  **do**
  - 3:   **for**  $steps = 1, 2, \dots, ratio$  **do**
  - 4:     unfreeze  $D_A, D_B$  weights,  $F, G$  weights
  - 5:     train  $D_A, D_B$  by maximizing final loss  $\mathcal{L}_{final}$  (minimizing  $-\mathcal{L}_{final}$ )
  - 6:   **end for**
  - 7:   freeze  $D_A, D_B$  weights, unfreeze  $F, G$  weights
  - 8:   train  $F, G$  by minimizing final loss  $\mathcal{L}_{final}$ , for SSDA including  $\mathcal{L}_{cc}$
  - 9: **end for**
- 

**A. Pseudo-Labeling**

**Pseudo-labeling** is a semi-supervised learning method: a neural network is trained on labeled source data by minimizing a supervised classification loss function. Then unlabeled data is used to improve generalization and accuracy on the target domain, hence further improving accuracy for source and, especially, for unlabeled target data. We assume softmax output unit, categorical cross entropy as a loss function  $\mathcal{L} = \mathcal{L}_{CE}$ . The goal is to build a statistical model  $f_{ada} : X \rightarrow Y$  so that the classification error is minimized for  $D_S$ . For labeled source data, we have the following optimization problem:

$$f_{ada}(\cdot) = \min_{\tilde{f}} \mathcal{L}(\tilde{f}(X_S), Y_S)$$

$$= \min_{\tilde{f}} \sum_{(x_S^i, y_S^i) \in S} \mathcal{L}(\tilde{f}(x_S^i), y_S^i). \quad (13)$$

where  $(x_S^i, y_S^i)$  is any paired source (input, output) sample drawn i.i.d from  $D_S$ .  $C$  represents the number of class labels,  $y^k$  is the categorical encoding of the label, and  $f^k$  is the network output, meaning the extended class probability for the  $k$ -th class label.

Pseudo-labels  $\hat{y}_T$  are classes assigned to unlabeled target samples by using a classifier trained on  $S = SL$  and selecting the class with maximum predicted probability:

$$\hat{y}_T^j = \operatorname{argmax} f_{ada}(x_T^j) \quad (14)$$

The process of generating pseudo-labels can be accomplished in a so called *online* and an *offline* manner:

- *Offline PL*: A classifier is trained with the available labeled data for a fixed amount of iterations. A probability threshold is defined. Then the classifier is used to generate pseudo-labels for unlabeled target data. Pseudo-labels with a softmax output higher than the threshold are selected, added to the labeled data, and used for the next training phase to further improve the accuracy of the classifier model, especially for target data samples.
- *Online PL*: The thresholding and pseudo-labeling are being performed in a temporary manner on each mini-batch during the classifier training. A loss weight based on the number of training steps is introduced to account for the lower classification accuracy, especially at the

beginning of the training. Increasing the weight during is recommended (see [14]). The loss function  $\mathcal{L}$  is defined as:

$$\mathcal{L} = \mathcal{L}(f_{ada}(X_S), Y_S) + \alpha(t)\mathcal{L}(f_{ada}(X_T), \hat{Y}_T), \quad (15)$$

where  $\alpha(t)$  is the weighting function schedule based on the total number of steps  $T$ .

**B. AdaMatch**

**AdaMatch** [3] exploits online pseudo-labeling and enriches it with consistency regularization, addressing the distribution shift between source and target domains present in the *batch norm statistics*, flexible pseudo-label confidence threshold, and modified version of distribution alignment. Let define  $f : X \rightarrow Y = R^C$  the classifier model as before, where  $f$  is composed as  $f(x) = g(h(x))$ . Specifically,  $h : X \rightarrow Z = R^C$   $f_1 : X \rightarrow Z := R^C$ , where  $Z$  represents the logits in output for each of the  $C$  classes. Instead,  $g$  is defined as the softmax operator, which allows  $f$  to map the input into the probability space.

Then, AdaMatch follows the following pipeline.

1) *AdaMatch - Image Augmentation*: Each minibatch  $X_{SL}, X_{TU}$  are augmented with both a *weak* and a *strong* augmentation,  $X_{SL}^{aug} = \{X_{SL,w}, X_{SL,s}\}$ ,  $X_{TU}^{aug} = \{X_{TU,w}, X_{TU,s}\}$  respectively. The logits of the augmented images are computed as:

$$\{Z'_{SL}, Z_{TU}\} = h_{ada}(X_{SL}^{aug}, X_{TU}^{aug})$$

$$Z''_{SL} = h_{ada}(X_{SL}^{aug}), \quad (16)$$

where  $Z'_{SL}$  and  $Z''_{SL}$  could be different because of the batch normalization statistics since they are obtained from differently assembled batches (respectively with and without the target domain images).

2) *AdaMatch - Random Logit Interpolation*: The logits  $Z'_{SL}$  and  $Z''_{SL}$  are interpolated:

$$Z_{SL} = \lambda Z'_{SL} + (1 - \lambda) Z''_{SL}, \quad (17)$$

where  $\lambda$  is sampled from a uniform distribution with range  $(0, 1)$  to weight each logit.

3) *AdaMatch - Distribution Alignment*: The idea is to encourage the target unlabeled distribution of pseudo-labels to follow the distribution of the source labeled data, assuming similar source and target label distributions. AdaMatch estimates the source label distribution from the output of the model on the labeled source data. Given the logits for weakly augmented source  $Z_{SL,w}$  and target  $Z_{TU,w}$  samples, the pseudo-labels are computed:

$$\hat{Y}_{SL,w} = \operatorname{softmax}(Z_{SL,w})$$

$$\hat{Y}_{TU,w} = \operatorname{softmax}(Z_{TU,w}).$$

Distribution alignment is applied by multiplying the target unlabeled pseudo-labels by the ratio between the expected value of the weakly augmented source pseudo-labels  $\mathbb{E}[\hat{Y}_{SL,w}]$  and the expected value of the weakly augmented target pseudo-labels  $\mathbb{E}[\hat{Y}_{TU,w}]$ :

$$\tilde{Y}_{TU,w} = \operatorname{normalize} \left( \hat{Y}_{TU,w} \frac{\mathbb{E}[\hat{Y}_{SL,w}]}{\mathbb{E}[\hat{Y}_{TU,w}]} \right), \quad (18)$$

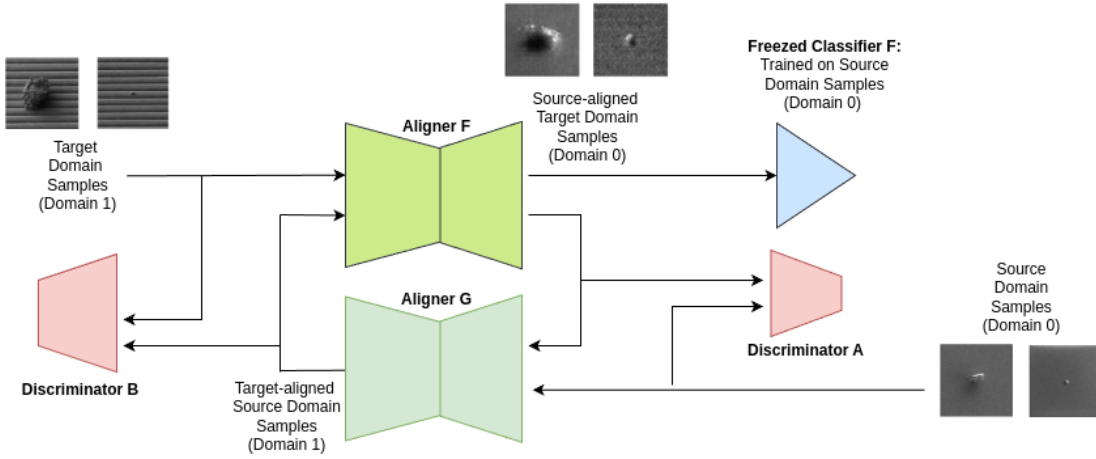


Fig. 2: Scheme of our approach based on DBACS and enhanced with the following modifications: cyclic adaptation, unpaired sample mapping, identity loss, feature matching, and multi-scale structure similarity. The classifier is trained on source domain data. The first generator (aligner  $F$ ) maps the target domain to the source domain to enable target data classification. Aligner  $F$  combined with aligner  $G$  are set up to enable bijective mapping within each domain. Discriminators are used for adversarial training. The discriminators and aligners compose the CycleGAN. The classifier, trained on the source domain, is frozen during the DBACS training. In this image example, Domain 0 is the Source Domain and Domain 1 is the Target Domain.

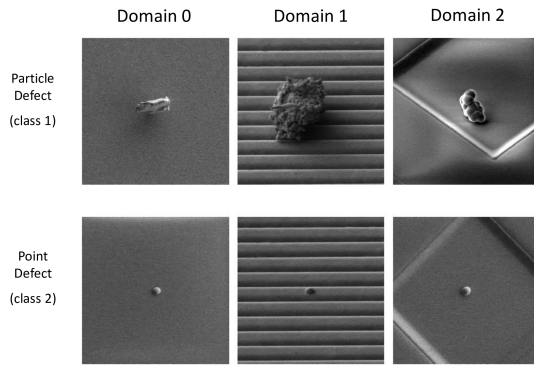


Fig. 3: In this study, we address the image classification problem in the context of the semiconductor sector considering two classes: particle and point defects. Each column represents a distinct domain. Our objective is to apply Domain Adaptation techniques, using one domain as the source (labeled data) and another as the target (unlabeled data). The objective is to achieve adaptation to the new domain without the need for expensive label collection phases.

where normalize makes the distribution sum to 1 again.

4) *AdaMatch - Loss Function*: The loss for the source labeled, both weakly and strongly augmented data, is defined as:

$$\mathcal{L}_{source} = \frac{\tau}{n_{SL}} \sum_{i=1}^{n_{SL}} \mathcal{L}(Y_{SL}^{(i)}, Z_{SL,w}^{(i)}) + \frac{\tau}{n_{SL}} \sum_{i=1}^{n_{SL}} \mathcal{L}(Y_{SL}^{(i)}, Z_{SL,s}^{(i)}) \quad (19)$$

The loss for the unlabeled, masked target data is defined as:

$$\mathcal{L}_{target} = \frac{\tau}{n_{TU}} \sum_{i=1}^{n_{TU}} \mathcal{L}(\text{stop\_grad}(\tilde{Y}_{TU,w}^{(i)}), Z_{TU,s}^{(i)}) \text{mask}^{(i)} \quad (20)$$

where  $\mathcal{L} = \mathcal{L}_{CE}$  is the categorical cross entropy loss and  $\text{stop\_grad}$  is a function that prevents gradient back-propagation on pseudo-labels, which is a standard practice in SSL that improves training. The final loss  $\mathcal{L}_{final}$  is then:

$$\mathcal{L}_{final} = \mathcal{L}_{source} + \mu(t)\mathcal{L}_{target} \quad (21)$$

where  $\mu(t)$  is a warm-up function.

5) *AdaMatch - Augmentation*: In AdaMatch, each source and target image is augmented both in a weak and a strong way. Each weakly augmented image is randomly horizontally mirrored with a certain probability. The strong augmentation is given by CTAugment [17] applied on top of the same horizontal flip. CTAugment is based on AutoAugment [36], it randomly selects a number of transformations for each sample and learns the magnitudes of each individual transformation on-the-fly. In particular, given a collection of transformations (augmentations), each sample of a mini-batch is augmented with a pipeline consisting of two transformations that are randomly and uniformly sampled; Cutout [18], by a square as big as 1/4 of the image size by 1/4 of the image size, with an area of 1/16 of the total area, is always an additional part of the strong augmentation pipeline and applied on top of the two augmentations. The magnitude of each transformation is first chosen randomly and then updated according to how close the model's predictions are to the true labels. The set of augmentations is given by: autocontrast, brightness, color, contrast, cutout, equalize, invert, identity, posterize, rescale, rotate, sharpness, shear\_x, shear\_y, smooth, solarize, translate\_x, translate\_y. Specific modifications of the augmentation procedure with respect to the original paper are as follows. In

order to avoid the introduction of new artifacts that are not present in the original data domain, only 180 degree rotations are applied (especially important for horizontal lines in domain 1). In addition, the size of the cutouts is selected small enough to avoid complete cutouts of any whole defect.

## VI. EXPERIMENTAL SETTING

### A. Dataset Description

The data described in this section and used in the experiments have been provided by Infineon Technologies AG; the dataset is composed of images taken by SEMs. As introduced in Section I, SEM is a type of electron microscope that produces images of a specimen by scanning its surface with a focused beam of electrons. The electrons, by interacting with atoms in the specimen, produce several informative signals about the surface topography and composition of the specimen. Secondary electrons emitted by the specimens' atoms excited by the electron beam can be detected by using an in-lens detector or an external detector. Depending on the type of detector, SEM images can therefore be divided into in-lens detector images and external detector images. In this work, both kinds of images are used.

The data includes different predefined *product technologies* that are characterized by different backgrounds on which the defects lie. Only defects are selected that can occur on multiple product technologies due to overlapping production steps. Therefore, a subset of the classes is chosen, and occurring defects are merged into two final classes, which are called *points* and *particle*. A particle defect is characterized by some kind of foreign particle, like, for example, dust, that is present on the current surface layer of the wafer. Point defects are impurities of the layer material and can disturb physical properties like, for example, the electrical conductivity of the wafer. Three different product technologies presenting three different image backgrounds are selected based on the availability of training data. The three domains are defined based on the corresponding product technologies: domain 0 has a plain, unstructured background; *domain 1* has horizontal lines in the background; *domain 2* exhibits square-like shapes of different sizes and locations in the background.

More in detail, all images are grayscale of shape 128x128, where for Domain 0, there are 1706 instances in the first class and 1516 instances in the other. Domain 1 has 639 and 503 instances in its respective classes. In domain 2, there are 577 instances in one class and 1196 in the other. It is also relevant to observe that the class distribution is balanced for domains 0 and 1 but imbalanced for domain 2.

### B. UDA and SSDA Scenarios

We are going to simulate unsupervised domain adaptation (UDA) as well as semi-supervised domain adaptation (SSDA). To simulate a UDA scenario, a source domain is selected and assumed to be fully supervised; hence, all available labels will be considered during training. One of the other domains is selected as the target domain, and all available labels are ignored and not used for training. For example, for UDA from domain 0 to 1, the labels of the samples from domain 0 are

considered, while those from domain 1 are ignored. In the SSDA scenario, a pre-defined percentage of samples (from both classes) from the defined target domain is labeled and available during the training phase. For evaluation of model accuracy, available labels in the corresponding test data sets are used.

In the UDA and SSDA scenarios, we evaluate and compare the following DA techniques: Offline and Online PL, AdaMatch, and DBACS. These results will also be compared against an *Oracle* (upper bound) and a *lower bound* (baseline). The Oracle represents the performance reported in Table II when trained on the source domain  $i$  and tested on the target domain  $i$ . Essentially, the Oracle is the model trained with the labeled data of the studied domain. The *lower bound* for a target domain  $j$  using the source domain  $i$  is defined by the values in Table II, where the model is trained on domain  $i$  and evaluated on domain  $j$ .

### C. Models

Several DA techniques are tested in this work, like Online and Offline PL, including also state-of-the-art AdaMatch and other techniques like DBACS. In this work, we examine deeply the different approaches by comparing them while also changing the underlying classifier architecture. Specifically, we test all the DA methods with the following three implemented architectures: MobileNet, Resnet50, Resnet101. We performed this additional examination to evaluate the robustness of DA methods with respect to architectural changes. Moreover, it is relevant to understand if the use of bigger networks like Resnet101 or lighter architectures like MobileNet can influence the final performance. All the models are implemented in Python by using Tensorflow<sup>2</sup>.

For all architectures, the pre-trained model from ImageNet is evaluated considering only the feature extractor part. Then, a binary classifier head suitable for the task at hand is attached, made by one dense layer. Since the network was pretrained on colored images (hence with a 3-channel input) and considering that the defect images are grayscale; it is more convenient to work with one channel only, a Conv2D layer with 3 filters was added on top as the first and new input layer.

The newly created layers are first trained for 30 epochs with the learning rate set to 0.0001 and Adam used as optimizer. While the first layers of the feature extractor remain frozen, the remaining layers are trained for 20 more epochs, with a learning rate that is 1/10 of the initial learning rate. Both phases of the training employ 20% of the training data for validation purposes; early stopping with patience 5 is applied on the validation loss.

### D. Hyperparameter Setting for the DA methods

All methods, unless diversely specified, are trained with the same hyperparameters as the classifier baselines.

1) *Offline and Online PL*: Both Offline and Online PL consider a confidence threshold  $\tau = 0.9$  and  $r = 3$  as unlabeled to labeled sample ratio. The offline ST considers  $N = 10$  iterations.

<sup>2</sup><https://www.tensorflow.org>

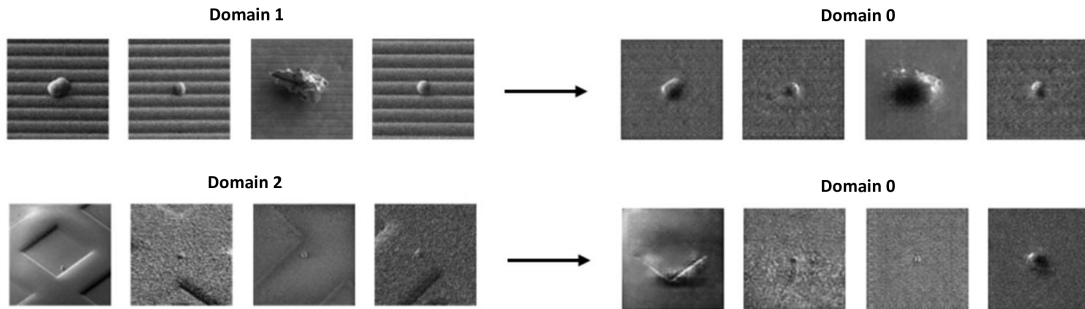


Fig. 4: Examples of defect images aligned to Domain 0 with DBACS in the UDA setting. The left column shows the original images; the right column shows the aligned images with changed backgrounds. The first row shows domain 1, and the second row shows domain 2.

TABLE II: **Baseline models accuracy.** The first column denotes the domain of the train data, and the remaining columns indicate the accuracy of the corresponding test set while changing the model. In **bold** the best value for each domain and model.

Training Domain	Test Domain 0			Test Domain 1			Test Domain 2		
	MobileNet	Resnet50	Resnet101	MobileNet	Resnet50	Resnet101	MobileNet	Resnet50	Resnet101
<b>0</b>	<b>0.976</b>	<b>0.921</b>	<b>0.981</b>	0.707	0.680	0.810	0.662	0.657	0.669
<b>1</b>	0.672	0.733	0.699	<b>0.973</b>	<b>0.858</b>	<b>0.982</b>	0.512	0.574	0.583
<b>2</b>	0.841	0.629	0.876	0.828	0.677	0.799	<b>0.959</b>	<b>0.811</b>	<b>0.971</b>
0 + 5% 1	94,06	95,66	94,52	88,71	90,82	92,84	72,55	82,53	84,54
0 + 5% 2	93,12	94,44	94,8	83,19	75,43	84,11	87,96	86,27	87,3

2) *AdaMatch*: For the AdaMatch approach, our results showed that blind usage of the default hyperparameters leads to suboptimal performance. Instead, tuning the data augmentations used and taking into account the effect of the different label distributions across domains can make it the most competitive approach for some source-target distributions. AdaMatch uses,  $\tau = 0.9$  and  $r = 3$  like the PL approaches. Then, it is used a weight decay of 0.001, a learning rate of 0.0002, and a batch size of 64. Given the limited amount of data, we reduce the number of steps by a factor of 8 with respect to the original paper. In order to compensate for the reduced training, the weight of the unsupervised loss is also adapted to prevent too much weight to the unsupervised loss too early in the training.

3) *DBACS*: The adversarial training of the other model parts of the architecture (discriminators and aligners) is carried out with a training ratio  $r_{adv} = 2$  in favour of the discriminators. The final batch size is  $64 * r_{adv} = 128$ , the learning rate  $lr = 0.00005$ , the optimizer is Adam, and the number of epochs is 300. The weights assigned to the different loss terms are  $\lambda_{ce} = 1.0$ ,  $\lambda_{adv} = 0.5$ ,  $\lambda_{cyc} = 0.3$ ,  $\lambda_{id} = 0.2$ ,  $\lambda_{fm} = 0.0$  and follow the indications in [34],

## VII. RESULTS

The DA methods and baselines are evaluated for the UDA setting in Section VII-A, while in Section VII-B a similar evaluation is performed for the SSDA setting.

### A. UDA scenario

In the unsupervised setting, we have access to the label for the source domain but not for the target.

As Oracle (optimal performance) is considered, the performance when the model is trained on the labeled domain. The results for each domain and for each model are reported in Table II. In general, we can observe that the best values are achieved by the ResNet101 (largest model) followed by the MobileNet (lightest model) and interestingly ending with the ResNet50. Therefore, while the largest model like ResNet seems to be preferred, satisfactory results can be achieved with lighter architectures like MobileNet.

When considering the *lower limit* as a baseline in terms of transfer learning, we consider the out-of-domain accuracy reported in Table II and as described in VI-B. When considering these values, it is interesting to note that both ResNet101 and MobileNet seem to have a discrete ability of generalization. Based on the considered source domain is normal to expect different performance in terms of domain adaptation. We report the results obtained by using as source the domain 0 in Table III, while in Table IV we report the results for target domains 0 and 2 using as source the domain 1.

The first general observation is that for the UDA setting, the DA techniques prove their usefulness in the semiconductor sector with SEM images by obtaining better performance than the lower limit (baseline). This first observation confirms that the use of DA techniques can be beneficial in practice when we don't have access to labeled data of the target domain or when it is too costly to obtain the labels of the new domain. The second observation is that the best DA approach depends on the chosen source domain and the best DA approach depends on the specific use case. In other words, the effectiveness of these methods can be highly dependent on the characteristics and distributions of the source data and all the tested approaches should be considered before choosing the

TABLE III: UDA models accuracy - source domain 0. In bold the best value for each domain and model.

Source → Target	0 → 1			0 → 2		
Model	MobileNet	ResNet50	ResNet101	MobileNet	ResNet50	ResNet101
lower limit	70,7	68	81	66,2	65,7	66,9
DBACS	79,3	72,4	80,2	64,53	61	65,64
Offline PL	83	<b>86</b>	85	55,32	52,78	53,72
Online PL	67,8	57	69,5	59,8	69,6	69,8
AdaMatch	<b>81,89</b>	84,15	<b>85,3</b>	<b>79</b>	<b>81,23</b>	<b>84,32</b>
Oracle	97,3	85,8	98,2	95,9	81,1	97,1

TABLE IV: UDA models accuracy - source domain 1. In bold the best value for each domain and model.

Source → Target	1 → 0			1 → 2		
Model	MobileNet	ResNet50	ResNet101	MobileNet	ResNet50	ResNet101
lower limit	67,2	73,3	69,9	51,2	57,4	58,3
DBACS	67,84	69,53	78,75	55,18	63,24	65,27
Offline PL	<b>85,78</b>	<b>91,38</b>	<b>88,79</b>	<b>82,64</b>	<b>85,64</b>	<b>87,65</b>
Online PL	72,3	74,51	76,01	71,44	73,56	72,04
AdaMatch	56,65	60,78	65,74	52,86	54,33	60,24
Oracle	97,6	92,1	98,1	95,9	81,1	97,1

TABLE V: SSDA (5% of labeled data in the target domain) models accuracy. In bold the best value for each domain and model.

Source → Target	0 → 1			0 → 2		
Model	MobileNet	ResNet50	ResNet101	MobileNet	ResNet50	ResNet101
lower limit	88,71	90,82	<b>92,84</b>	<b>87,96</b>	86,27	87,3
DBACS	81,67	82,74	82,86	70,03	77,87	79,83
Offline PL	<b>89,22</b>	<b>91,4</b>	92	87,68	83,95	<b>89,09</b>
Online PL	87,4	88,35	89,44	82,45	<b>87,68</b>	88,78
AdaMatch	88,1	89,11	90,12	85,25	86,75	87,13
Oracle	97,3	85,8	98,2	95,9	81,1	97,1

best one.

Another general observation concerns the application for the first time of the DBACS approach in the Computer Vision field (after all the modifications (as indicated in Section IV-A)) concerning the original approach to improve the results. More in detail, concerning the DBACS approach, in Figure 4, examples of defect images from domains 1 and 2 using as the source domain 0. Based on the results, we can note how the approach tries to change the data from the unlabeled domain into an already learned representation (domain 0). DBACS demonstrates superior performance compared to the lower limit (baseline), highlighting its relevance to be used in future studies in the Domain Adaptation field. Another advantage of the DBACS approach is that it remains unaffected by the source or target distribution, in contrast to the AdaMatch approach, which, in the case of relevant differences between the two distributions, could lead to low performance. However, it should be noted that DBACS is more computationally demanding compared to AdaMatch, especially given its complexity in terms of both architecture and optimization.

Examining the results from source domain 0 in Table III, it is evident that AdaMatch emerges as the top-performing approach. However, other methods such as DBACS and Offline PL, also exhibit strong performance, notably surpassing the lower limit (baseline) by a considerable margin.

In contrast, when observing the results from Table IV concerning the source domain 1, we can see that most of the DA approaches struggle to achieve performance comparable

to the Oracle (upper bound). This includes the AdaMatch and DBACS methods. However, the best performer is offline PL, showing a significant improvement over the lower limit. For instance, in the case of  $1 \rightarrow 0$  using ResNet 101, we achieve an accuracy of 88.79% with Offline PL compared to 69.9% with the lower limit. This can be explained by the fact that more data are pseudo-labeled after each iteration. This means that the model is more confident about its predictions on unlabeled target domain data as new target domain data is available for model training.

### B. SSDA scenario

In the SSDA scenario (a small part of the target domain data is available), the label availability improves the inter-domain performance of the DA models significantly (see Table V). This implies, in general, the importance of having even a small set of labeled data when possible. However, at the same time, it hardly improves the performance concerning the "lower limit" baseline. This raises concerns about the effectiveness of domain adaptation (DA) techniques in the semi-supervised domain adaptation (SSDA) scenario for the SEM images in the semiconductor manufacturing field, while, in contrast, for the UDA scenario, we can observe a large benefit in the application of DA techniques. Also, in this case, the best DA approach is Offline PL, though in some cases, the lower limit has better results.

### VIII. CONCLUSION AND FUTURE WORK

Training models able to generalize to out-of-distribution data can be a complex task. In fact, models trained on data coming from one domain struggle to generalize to other domains, where labeled data is not accessible. In this work, Pseudo-labeling techniques and generative adversarial methods for the Domain Adaptation field are applied for the first time in the semiconductor manufacturing field for SEM images. Different approaches are adopted and applied: the generative adversarial one represented by DBACS, which is meant to transform respective aligned input data from one domain to another, or the pseudo-label-based one, such as the AdaMatch model, is implemented and specified. The results obtained show how the DA techniques are able to work effectively in the unsupervised setting (UDA), with large improvements over the baselines and results that are not far from the oracle's performance. Moreover, from this analysis, it is observed that the best DA approach depends on the chosen source domain and the specific use case. In other words, the effectiveness of these methods can be highly dependent on the characteristics and distributions of the source data, and all the tested approaches should be considered before choosing the best one. In addition, the initial tests conducted in the Computer Vision field using the DBACS approach appear promising, demonstrating the method's applicability and suggesting its potential for future studies in the Domain Adaptation field, especially in the case of distribution misalignments between the source and target distribution.

As future research, we envision a focus on studying classification problems with more than two classes by including more defect classes, the impact of a larger amount of data, both labeled and unlabeled, and also on studying a multi-target domain adaptation scenario.

#### ACKNOWLEDGMENT

Infineon Technologies AG is gratefully acknowledged for the financial support of this research.

#### REFERENCES

- [1] M. Maggipinto, A. Beghi, and G. A. Susto, "A deep convolutional autoencoder-based approach for anomaly detection with industrial, non-images, 2-dimensional data: A semiconductor manufacturing case study," *IEEE Transactions on Automation Science and Engineering*, 2022.
- [2] D. Dalle Pezze, C. Masiero, D. Tosato, A. Beghi, and G. A. Susto, "Formula: A deep learning approach for rare alarms predictions in industrial equipment," *IEEE Transactions on Automation Science and Engineering*, 2021.
- [3] B. Roelofs, D. Berthelot, K. Sohn, N. Carlini, and A. Kurakin, "Adamatch: a unified approach to semi-supervised learning and domain adaptation," in *Proc. of the Int. Conf. on Learning Representations (ICLR)*, 2022.
- [4] N. Gentner and G. A. Susto, "Heterogeneous domain adaptation and equipment matching: Dann-based alignment with cyclic supervision (dbacs)," *Computers & Industrial Engineering*, vol. 187, p. 109821, 2024.
- [5] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *The journal of machine learning research*, vol. 17, no. 1, pp. 2096–2030, 2016.
- [6] M.-J. Wu, J.-S. R. Jang, and J.-L. Chen, "Wafer map failure pattern recognition and similarity ranking for large-scale data sets," *IEEE Transactions on Semiconductor Manufacturing*, vol. 28, no. 1, pp. 1–12, 2014.
- [7] L. Lorenti, D. Dalle Pezze, J. Andreoli, C. Masiero, N. Gentner, Y. Yang, and G. A. Susto, "Predictive maintenance in the industry: A comparative study on deep learning-based remaining useful life estimation," in *2023 IEEE 21st international conference on industrial informatics (indin)*. IEEE, 2023, pp. 1–9.
- [8] K. Imoto, T. Nakai, T. Ike, K. Haruki, and Y. Sato, "A cnn-based transfer learning method for defect classification in semiconductor manufacturing," in *2018 international symposium on semiconductor manufacturing (ISSM)*. IEEE, 2018, pp. 1–3.
- [9] T. Schlosser, F. Beuth, M. Friedrich, and D. Kowerko, "A novel visual fault detection and classification system for semiconductor manufacturing using stacked hybrid convolutional neural networks," in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2019, pp. 1511–1514.
- [10] S. Cheon, H. Lee, C. O. Kim, and S. H. Lee, "Convolutional neural network for wafer surface defect classification and the detection of unknown defect class," *IEEE Transactions on Semiconductor Manufacturing*, vol. 32, no. 2, pp. 163–170, 2019.
- [11] J. O'Leary, K. Sawlani, and A. Mesbah, "Deep learning for classification of the chemical composition of particle defects on semiconductor wafers," *IEEE Transactions on Semiconductor Manufacturing*, vol. 33, no. 1, pp. 72–85, 2020.
- [12] K. S.-M. Li, X.-H. Jiang, L. L.-Y. Chen, S.-J. Wang, A. Y.-A. Huang, J. E. Chen, H.-C. Liang, and C.-L. Hsu, "Wafer defect pattern labeling and recognition using semi-supervised learning," *IEEE Transactions on Semiconductor Manufacturing*, vol. 35, no. 2, pp. 291–299, 2022.
- [13] D.-H. Lee *et al.*, "Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks," in *Workshop on challenges in representation learning, ICML*, vol. 3, no. 2, 2013, p. 896.
- [14] K. Sohn, D. Berthelot, N. Carlini, Z. Zhang, H. Zhang, C. A. Raffel, E. D. Cubuk, A. Kurakin, and C.-L. Li, "Fixmatch: Simplifying semi-supervised learning with consistency and confidence," *Advances in Neural Information Processing Systems*, vol. 33, pp. 596–608, 2020.
- [15] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," *Advances in neural information processing systems*, vol. 30, 2017.
- [16] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan, "Augmix: A simple data pro-

- cessing method to improve robustness and uncertainty,” *ArXiv*, vol. abs/1912.02781, 2020.
- [17] D. Berthelot, N. Carlini, E. D. Cubuk, A. Kurakin, K. Sohn, H. Zhang, and C. Raffel, “Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring,” *arXiv preprint arXiv:1911.09785*, 2019.
- [18] T. DeVries and G. W. Taylor, “Improved regularization of convolutional neural networks with cutout,” *arXiv preprint arXiv:1708.04552*, 2017.
- [19] A. Mottaghi, M. A. Jamal, S. Yeung, and O. Mohareri, “Adaembed: Semi-supervised domain adaptation in the embedding space,” 2024. [Online]. Available: <https://arxiv.org/abs/2401.12421>
- [20] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, “Image-to-image translation with conditional adversarial networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.
- [21] M. Mirza and S. Osindero, “Conditional generative adversarial nets,” *arXiv preprint arXiv:1411.1784*, 2014.
- [22] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” *Advances in neural information processing systems*, vol. 27, 2014.
- [23] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, “Unpaired image-to-image translation using cycle-consistent adversarial networks,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.
- [24] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko, “Simultaneous deep transfer across domains and tasks,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4068–4076.
- [25] B. Sun and K. Saenko, “Deep coral: Correlation alignment for deep domain adaptation,” in *European conference on computer vision*. Springer, 2016, pp. 443–450.
- [26] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada, “Maximum classifier discrepancy for unsupervised domain adaptation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3723–3732.
- [27] Z. Li, B. Wan, C. Mu, R. Zhao, S. Qiu, and C. Yan, “Ad-aligning: Emulating human-like generalization for cognitive domain adaptation in deep learning,” in *2024 5th International Conference on Electronic Communication and Artificial Intelligence (ICECAI)*. IEEE, May 2024, p. 794–798. [Online]. Available: <http://dx.doi.org/10.1109/ICECAI62591.2024.10675013>
- [28] J. Kataoka and H. Yoon, “Avatar: Adversarial self-supervised domain adaptation network for target domain,” 2023. [Online]. Available: <https://arxiv.org/abs/2305.00082>
- [29] N. Gentner, M. Carletti, A. Kyek, G. A. Susto, and Y. Yang, “Dbam: Making virtual metrology/soft sensing with time series data scalable through deep learning,” *Control Engineering Practice*, vol. 116, p. 104914, 2021.
- [30] Y. Ding, L. Sheng, J. Liang, A. Zheng, and R. He, “Proxymix: Proxy-based mixup training with label refinery for source-free domain adaptation,” *Neural Networks*, vol. 167, pp. 92–103, 2023.
- [31] J. Li and H. Sun, “Nacl: noise-robust cross-domain contrastive learning for unsupervised domain adaptation,” *Machine Learning*, vol. 112, no. 10, pp. 3473–3496, 2023. [Online]. Available: <https://doi.org/10.1007/s10994-023-06343-8>
- [32] J. Li, G. Li, and Y. Yu, “Inter-domain mixup for semi-supervised domain adaptation,” *Pattern Recognition*, vol. 146, p. 110023, Feb. 2024. [Online]. Available: <http://dx.doi.org/10.1016/j.patcog.2023.110023>
- [33] J. An, C. Zhao, and D. Wu, “Semi-supervised generalized source-free domain adaptation (ssg-sfda),” in *2023 International Joint Conference on Neural Networks (IJCNN)*, 2023, pp. 1–8.
- [34] A. Gokaslan, V. Ramanujan, D. Ritchie, K. I. Kim, and J. Tompkin, “Improving shape deformation in unsupervised image-to-image translation,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 649–665.
- [35] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.
- [36] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, “Autoaugment: Learning augmentation policies from data,” *arXiv preprint arXiv:1805.09501*, 2018.

**Adrian Poniatowski** Adrian Poniatowski holds a Master’s degree in Artificial Intelligence from the University of Bologna, where he conducted his thesis in collaboration with Infineon Technologies.

**Natalie Gentner** She is currently a product engineer at Infineon Technologies AG (Neubiberg, Germany). She earned her PhD in Information Engineering from the University of Padova and owns a M.Sc in Mathematics from the Technical University in Munich.

**Manuel Barusco** Manuel Barusco received the M.S. degree in Computer Engineering in 2023. He is currently pursuing a Ph.D. degree in Computer Science at the University of Padova, Italy. His focus is in the design of ML and Deep Learning approaches for anomaly detection, domain adaptation and other Industry 5.0 related tasks.

**Davide Dalle Pezze** He is currently a Postdoctoral Researcher at the University of Padova (Italy), specifically at the Department of Information Engineering. His research interests focus on (i) Continual Learning, and (ii) Industry 4.0 topics, including Predictive Maintenance, Fault Detection, and Visual Anomaly Detection.

**Samuele Salti** Samuele Salti is currently an Associate Professor with the Department of Computer Science and Engineering (DISI), University of Bologna, Italy. His main research interests include computer vision and machine learning. He has coauthored 53 publications and eight patents. In 2020, he co-founded the start-up eyecan.ai.

**Gian Antonio Susto** He is currently an Associate Professor at the University of Padova. He held visiting positions at the University of California, San Diego, the National University of Ireland, Maynooth, and Infineon Technologies Austria AG, Villach, Austria. He is also one of the Co-Founder of the Machine Learning Company Statwolf. He is the author of more than 100 publications in peer-reviewed venues.