UCLouvain

École polytechnique de Louvain

Interpretability of Deep Unsupervised Domain Adaptation

Analysis of distribution shift and critical comparison of methods

Author: **Gatien Dony**
Supervisors: **Laurent Demanet, Laurent Jacques, Estelle Massart**
Reader: **Cyril De Bodt**
Academic year 2023–2024
Master [120] in Mathematical Engineering

Acknowledgements

I would like to express my deepest gratitude to my three supervisors, Prof. Laurent Demanet, Prof. Laurent Jacques, and Prof. Estelle Massart for their guidance, expertise, and precious advice throughout this work. Especially, I want to thank Prof. Demanet for having proposed this fascinating topic and welcomed me into his department. And I thank Prof. Jacques and Prof. Massart for having accepted to co-supervise this work. Their availability and kindness have been a great support for me.

Also, this work would not have been the same without the help of Borjan Geshkovski in the early stages of the project. Thanks to Matthew T.C. Li and Olivier Leblanc for the insightful discussions they provided me with. I would like to thank Cyril de Bodt for having accepted to evaluate this work.

I thank everyone who supported me during my studies, especially Victoria, for her constant support and love since the beginning.

Computational resources have been provided by the supercomputing facilities of the Université catholique de Louvain (CISM/UCL) and the Consortium des Équipements de Calcul Intensif en Fédération Wallonie Bruxelles (CÉCI) funded by the Fonds de la Recherche Scientifique de Belgique (F.R.S.-FNRS) under convention 2.5020.11 and by the Walloon Region.

Working on Deep Learning with Deep Learning

Following the rules of UCLouvain about assisted writing, I declare that this work (thesis and code) has been written with the help of the autocompletion tool of GitHub Copilot. Everything has been verified and corrected by myself, and no important part has been artificially generated.

Contents

Tout est affaire de décor.

Bierstube, L. Aragon

1 Introduction

One could summarize Machine Learning (ML) with a few words by stating that a machine may be able to learn a certain task if we feed it with a sufficient amount of data. In the specific case of supervised ML, the machine aims at predicting an outcome value from an income, *e.g.* predicting the price of a house from its characteristics or identifying a handwritten digit. Very intuitively, the data it will learn from at the training phase must be similar enough to the data on which it will be used later, at the inference phase. How could a computer estimate the price of a house in Boston given solely the characteristics and prices of houses in Louvain-La-Neuve? To express that condition with mathematical concepts, the datasets should be sampled from the same distribution.

When this assumption cannot be met and our model is constrained to predict an unseen distribution, we say that there is a *distribution shift* (DS). Then, we are now dealing with two distributions: the source distribution, which is freely available (in terms of the number of examples and completeness of the learning pairs), and the target distribution from which the data met at inference time will be sampled. To really face the problem of DS, we must consider cases where the target is partially or totally unknown.

The concept of distribution shift is very broad and can be encountered in many applications. This work will focus on a specific instance of shift : *the domain adaptation problem* (DA). To transpose it to our introductory example, let us think of a Bostonian real estate expert who disposes of historical data from his Louvanist colleagues (being our source) on one side and only house descriptions by his clients (being our target) on the other side. He will endeavor to extract as much information as possible from the data within reach to predict prices. Domain adaptation is this precise situation. We dispose of a labeled source dataset and an unlabeled target dataset. Another way to see it is to leverage unlabeled data to improve the performance of a model trained on shifted data, to *adapt* the model to a new domain.

As we said, domain adaptation can find applications in many fields. And finding methods able to cope with this type of DS is of prime interest. From its first introduction, that can be traced back to 2000 [Shimodaira, 2000], the field has gained a growing interest. Naturally, researchers tried to formalize it with learning theory to better understand the problem and find efficient solutions. And as with every machine learning framework, it has been strongly influenced by the emergence of deep learning. To testify to its importance and openness, NeurIPS 2023 has dedicated a benchmark on the DA problem (WILDS 2.0 UDA Benchmark [Sagawa et al., 2022]).

The goal of this work is to provide an overall view of the possible solutions to the domain adaptation problem. Therefore, the fundamental interest of this thesis is the methods from a theoretical and practical point of view. More specifically, we'll give a try at understanding how these techniques should tackle the shifts, what their limitations are and how possible it is to overcome them. Following our intuitive example, adapting the model from Louvain-La-Neuve to Brussels is probably much easier than from Louvain-

La-Neuve to Boston. It probably means there is an underlying dependency on the notion of distance between domain. What kind of distance, and how do the methods take it into account? All these questions and even more will be addressed and discussed through practical and controllable experiments. More precisely, we will unify theoretical results to derive fundamental criteria for the success of DA methods. This will lead us to the definition of impossible cases for which we can never ensure success. The methods will be presented in terms of how they fit these criteria. And last but not least, we will provide numerical experiments on artificial datasets suffering from controllable shifts to support theory and observe practical behaviors.

Before diving into the details, we will need to take some time to clearly define the problem that we consider. Indeed, the DS and DA literature may take the wind out of the newcomer's sail with the number of variants and subtleties in the types of problems. Moreover, the terminology of DA lacks unification. As said in [Lemberger and Panico, 2020], *let's clear up the fog* of DA by clearly exposing the ramifications and stating the ones we focus on.

1.1 Distribution shifts

To introduce some notations, we will have input data from a set $\mathcal{X}$ and output data from a set $\mathcal{Y}$. The mapping of interest is denoted by $f : \mathcal{X} \to \mathcal{Y}$. On the one hand, the source distribution is denoted by $\mathcal{S}$ and the target distribution by $\mathcal{T}$. We say that there is a distribution shift if $\mathcal{S} \neq \mathcal{T}$. One fundamental idea throughout this work is that this condition is not very relevant as such because two distributions may be very close to each other but yet not equal. To better define differences between source and target distributions, we can define different types of shifts [Farahani et al., 2021].

1. **Covariate shift** This is the most common notion of DS. The inputs follow different distributions in source and target domains, $\mathcal{S}_X \neq \mathcal{T}_X$, but the conditional probability of the output given the input is conserved across source and target domains, $\mathcal{S}_{Y|X} = \mathcal{T}_{Y|X}$.

2. **Prior shift** In this case, the conditional probability of the output data given the input data remains the same $\mathcal{S}_{Y|X} = \mathcal{T}_{Y|X}$ but the prior varies, $\mathcal{S}_Y \neq \mathcal{T}_Y$.

3. **Concept shift** This type of shift concerns cases in which the distribution of the input is domain-invariant $\mathcal{S}_X = \mathcal{T}_X$ but the conditional probability of the output given the input varies, $\mathcal{S}_{Y|X} \neq \mathcal{T}_{Y|X}$.

For the rest of this work, we will only take into account the covariate shift scenario. We will see later that this type of shift can be subdivided. This is probably the most investigated type of shift. Also, it is the only type of shift that can be tackled with unsupervised domain adaptation (detailed below). The covariate shift will also be an important notion for the theoretical part of this work. Indeed, it is more than a convenient hypothesis but a key property for the success of DA.

1.2 Domain adaptation

To the problems of distribution shift, the methods aiming at leveraging the partial knowledge we have on the target distribution to improve performance in the latter domain are

called domain adaptation methods. In other words, DA is the attempted solution to a sub-class of two-domains distribution shift problem. Depending on what we can sample from the target distribution, we define different types of domain adaptation. This classification of DA is one among many others, but probably the most important one. Other subtleties will be exposed afterwards.

1. **Supervised domain adaptation (SDA)** Here, we can access to a few complete learning pairs of the target distribution. However, the number of samples is too small to train the whole model from the target only.

2. **Unsupervised domain adaptation (UDA)** In this case, we only have access to half of the learning pairs, *i.e.* the input data on target domain.

3. **Semi-supervised domain adaptation (SSDA)** This is a mix of the two previous cases. We have access to a few complete learning pairs and to input data.

The focus of this work will be on unsupervised domain adaptation, which will be simply referred to as domain adaptation. This can be summarized by the following. We consider two sets of samples generated by :

$$y_S = f(x_S) \text{ where } x_S \sim \mathcal{S}_X$$
$$y_T = f(x_T) \text{ where } x_T \sim \mathcal{T}_X$$

The data to which we have freely access is in blue and the data on which we will have to make predictions is in red. We do not make any further assumptions on the nature of the mapping f yet. In general, it will be stochastic and non-linear. It is also important to underline that this mapping is not known and that the goal of DA is to estimate it from the source data and then use it to make predictions on the target data.

Mixing this notion of UDA with the problem of covariate shift exposed before is the most investigated case of DA/DS. Also, it represents probably the most challenging. Indeed, we will look for models able to perform estimation on totally unknown data at first sight. In some sense, that can be compared to notions of extrapolation. To motivate the reader, we present below a few examples of situations where unsupervised domain adaptation could be useful. All of them would normally respect the covariate shift assumption.

- **Noisy classification/regression** : The source dataset is composed of clear instances (images, sounds, *etc.*) and the target dataset is composed of noisy instances. Of course, the model could suffer from unseen noise at inference time.

- **Synthetic to real** : Obtaining a cleanly labeled dataset is often an exhausting or nearly impossible process. Therefore, one could be tempted to generate synthetic data (*e.g.* with generative deep learning). However, the model trained on this data may suffer from incontrollable bias that could be correct by DA.

- **Numerical simulations to real for inverse problem** : When considering inverse problems, domain adaptation could also be useful. On one side, we can generate

synthetic data with numerical approximations of the forward operator. On the other side, obtaining a dataset with labels (hidden state in this case) is often impossible, while obtaining unlabeled data (observations) is much more feasible.

- **Operator shift in inverse problems** : In the context of inverse problems, one may have access to a labeled dataset with a certain forward operator and then, have to make predictions with another one. This corresponds to an unsupervised domain adaptation problem.

- **Online training with user data** : As the use of deep-learning-powered tools grows, we could think of the following situation : the tool is trained before its release on a large dataset (for object or sound recognition for example). Then, the user will use it, and expose it to new data asking for inference. Our model could store these unlabeled samples to adapt to the user's data.

To provide a broader view of the problem, we can identify domain adaptation as a subset of the general paradigm of transfer learning (TL). This very active field of research tries to find ways to extract learned components from a machine learning model to re-use it in another context.

Using vocabulary from TL, DA is a specific case of *homogeneous* TL, which means that data modalities are similar across the domains. TL is a very broad field, and not all its definitions are consensual. By metonymy, TL is sometimes used to refer to *task-oriented* transfer learning while DA names *domain-oriented* transfer learning. Figure 1 from [Pan and Yang, 2010] shows the usually admitted subdivision of fields in TL.

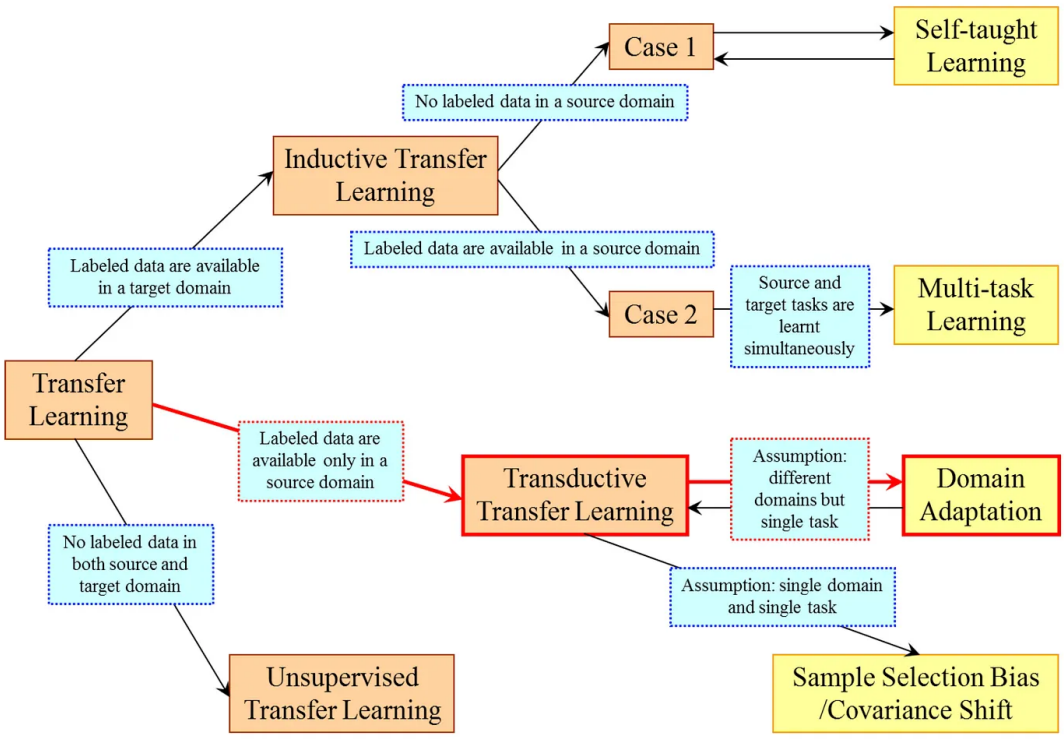


Figure 1: Ramification of transfer learning. From [Pan and Yang, 2010]

Furthermore, DA can also be split into several subfields depending on what assumptions are made and what types of methods are preferred. This is briefly summarized in figure 2. The definition of DA made before already make some of these assumptions, but it is almost impossible to provide a clear definition of DA without assuming some subtleties. Nevertheless, one can observe that inside the very broad field of TL, DA in itself is quite large and that the ideas covered in this work are only a part of it.

The notions of supervised/unsupervised/semi-supervised have already been introduced before. The other criteria are the following.

1. **Source available/Source-Free** One can see source-free DA as a very specific case that is sufficiently different from the general framework. There, source data is not directly available. We solely dispose of the model trained on the latter.

2. **Closed/Partial/Open DA** This is most often defined for classification tasks. However, we can derive a similar condition for regression. In closed DA, the set of classes is the same in both domains. Partial DA refers to the case in which the target domain has fewer classes than the source domain. Open DA is the most general case. The target domain has classes that are not present in the source domain. Regarding regression, the idea of sets of classes can be replaced by the support of the distributions of outcomes.

3. **Single/Multi-source DA** Even if it is less explored than single-source DA. Multi-Source DA considers that we dispose of several source domains.

A first naive approach in this setting would be to regress a model on the source data and then feed it with the target inputs x_T. However, it would be a pity not to exploit available knowledge on the target distribution. DA can be seen as a way to leverage the incomplete knowledge of the target distribution to improve performance in this domain. To measure the performance of a DA model, the difference in performance on the target domain with the baseline model with a similar architecture will be crucial.

In this work, we will focus on the case of unsupervised domain adaptation as it can be seen as the most challenging case. Then, the shifts encountered will be covariate shifts. To enhance the versatility of the results and leveraging the performance of deep learning models, we will focus on deep domain adaptation methods which means that the techniques will imply neural network architectures.

Even if they share some aspects regarding methods and theory, domain adaptation should not be confused with Domain Generalization (DG) [Ahuja et al., 2021, Xu and Raginsky, 2017, Nguyen et al., 2021b]. In this context, there are several source domains that are totally available and the objective is to infer in target domains that are unknown at training. One concept is not a generalization of the other : in DA we dispose of half knowledge (the inputs) on the domain of interest but only complete learning pairs on one domain, in DG we do not have access to anything from the target domain but complete pairs on several domains.

1.3 State of the art

As exposed in the introduction, from the very general idea of machine learning the question of generalization abilities of models arises quickly. And with that comes the

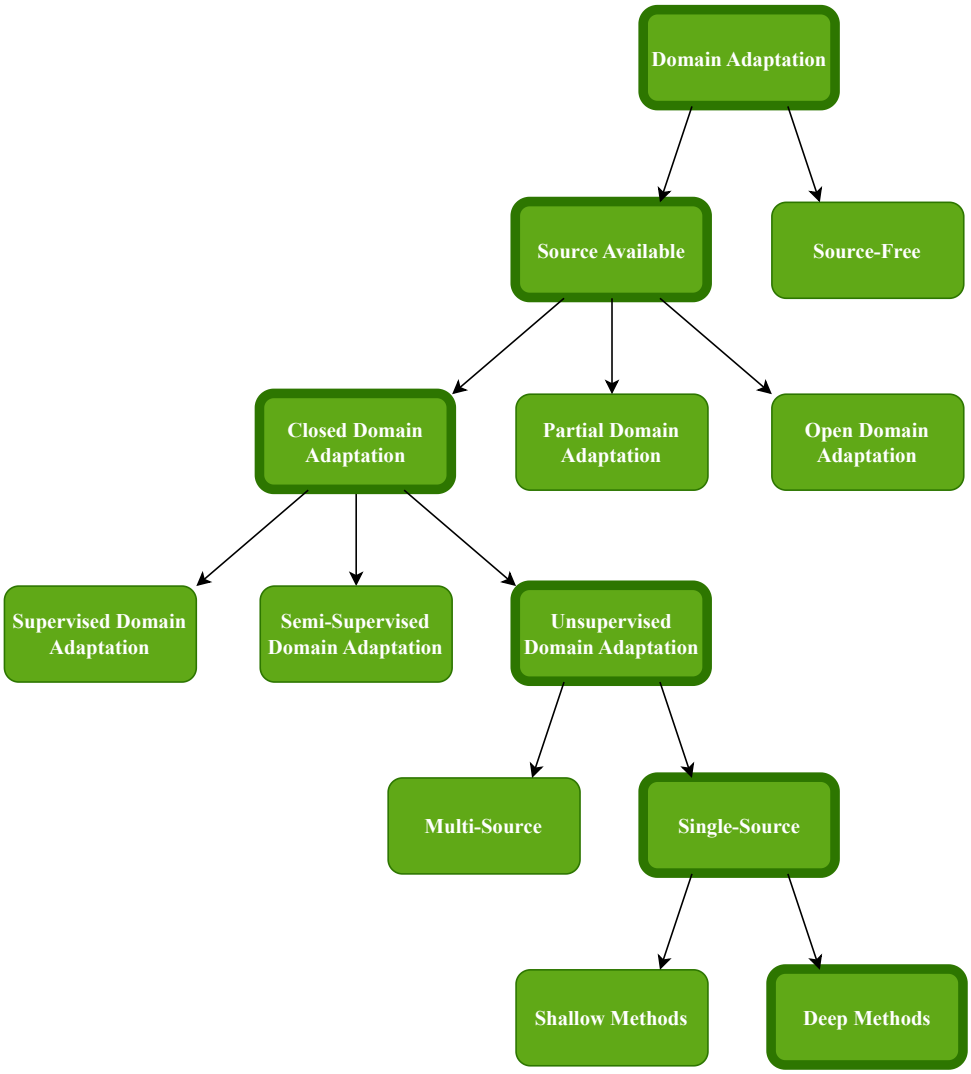


Figure 2: Considered ramification of DA. Aspects considered in this work are framed.

idea of distribution shifts. Therefore, the problem of handling them is not new. One of the first works recognized as treating domain adaptation is the one by Shimodaira [Shimodaira, 2000]. In this paper, the author proposes a method to improve the accuracy of an estimator facing covariate shift. At the time, the name domain adaptation was not yet cited. A bit later, [Roark and Bacchiani, 2003, Ando, 2004] were addressing similar problems in the context of Natural Language Processing (NLP). The concept of domain adaptation and its name have been introduced by the seminal work [Ben-David et al., 2006]. Moreover, this paper introduced the theoretical notions concerning domain adaptation. All these first ideas of domain adaptation were applied to NLP problems. Then, computer vision entered the picture. This was introduced in the works of Saenko and Darrel [Saenko et al., 2010]. First, shallow methods were used on features discovered by feature extractors, then the DA techniques were directly inserted into deep architectures [Csurka, 2017]. Numerous surveys have been written to gather the different methods [Farahani et al., 2021, Kouw and Loog, 2019, Kouw and Loog, 2021, Lemberger and Panico, 2020, Wilson and Cook, 2020, Liu et al., 2022, Wang and Deng, 2018,

Guan and Liu, 2022, Ramponi and Plank, 2020].

Like transfer learning in general, domain adaptation is made up of dozens of different methods. Some of them are very close or extensions of previous ones. Some of them rely on totally different principles. Consequently, each survey proposes a classification of the methods that is not always compatible with the others. The methods presented here below are probably the most representative of how DA has been approached and the most general ones.

Historically, before the advent of deep learning, the first methods used to tackle distribution shift were shallow ones. The latter rely mostly on refactoring the covariates or the labels. As the emphasis of this thesis is not on shallow models, we will not go into details about these methods.

The first one historically and in terms of simplicity is the importance weighting [Shimodaira, 2000]. The global idea is to give more importance in the learning process to samples from the source distribution that are similar to the target one. To a sample $x_i^S \sim \mathcal{S}_x$, a weight $w_i = \frac{\mathcal{T}_x(x_i)}{\mathcal{S}_x(x_i)}$ is associated. This specific choice of weights is motivated theoretically as it corresponds to the Radon-Nikodym derivative of the target distribution with respect to the source distribution. To obtain the weights, there are two possible approaches : estimate both marginals or directly estimate the ratio. The first one requires estimating distributions from data (either parametrically or non-parametrically). For the second choice, there are several alternatives like Kernel Mean-Matching (KMM)[Huang et al., 2006] or Kullback-Leibler Importance Estimation Procedure (KLIEP) [Sugiyama et al., 2007]. Both of these methods are based on minimizing a metric between $w\mathcal{S}_x$ and $\mathcal{T}_x$.

Afterwards, another class of shallow methods is based on projecting the samples onto subspaces and then, aligning them. The main instances of this class are subspace Alignment (SA) [Fernando et al., 2013], Sampling Geodesic Flow (SGF) [Gopalan et al., 2011] and its extension with kernel embeddings, Geodesic Flow Kernel (GFK) [Gong et al., 2012].

As the previous type of methods aimed at finding a common representation of the data from both domains, the next one, transformation-based methods tries to discover a transformation between domains. Once the latter was obtained, one could generate synthetically labeled target data by transforming source samples. For example, if one wants to perform object recognition with a source domain made of sketches and target, of pictures, one could perform sketch-to-image transformation on the input data of the source domain and then train any model on this modified data-set. In this category, there are mainly two methods. The well-known Correlation Alignment (CORAL) [Sun et al., 2016] transforms the source data to have the same correlation matrix as the target data. The second approach is related to the Optimal Transport (OT) problem which will be presented afterwards. Without going into details, the idea is to find a transformation that minimizes a chosen cost between to distributions. The method is called Optimal Transport-based Domain Adaptation (OTDA) [Courty et al., 2017b].

Then, deep learning architectures specifically designed for DA have been developed. In a supervised DL context, the inputs of neural networks are modified by a sequence of non-linear transformations yielding intermediate representations of the data to finally obtain the expected output. Details will be provided later (in section 4), but the global idea of most DeepDA techniques relies on finding domain-invariant representations that are still informative with respect to the required task. Ways to fulfill these two objectives

are numerous and generate a wide variety of methods.

About the theory of DA, most of it is summarized in [Redko et al., 2019]. Globally, early theoretical results have been established for simplified cases (binary classification, linear models, idealized features spaces) but most of them have never been extended to the general case. Also, the deep learning framework is not rich with theoretical results. PAC-Bayesian aspects have been studied in [Redko et al., 2019]. Recently, an interesting piece of theory has been notices in [Johansson et al., 2019], it exposes the existence of impossible domain adaptation problems and emphasizes the importance of support overlap over equality of distributions. The existence of pathological cases has also been studied in [Zhao et al., 2019] and [Rosenfeld et al., 2021]. This consideration has been derived from generalization failure as presented in [Beery et al., 2018]. This framework finally lead to the linking between Deep Learning and the information bottleneck principle. The generalization abilities of Deep Learning methods have been interestingly linked with the information bottleneck framework in [Kirsch et al., 2021]. An interplay between invariant risk minimization (that is also related to out-of-distribution generalization) and the information bottleneck principle has been studied in [Ahuja et al., 2021].

Testing the robustness of DA methods against adversarial attacks have been investigated in [Wang et al., 2024].

About inverse problems is clearly not the most investigated field in DA. However, we can find some works that have been done in this direction. We cite [Singhal and Majumdar, 2020] that is probably the most general one. The authors of [Liu et al., 2023] present an applied result of DA on computational imaging. Finally, the work of [Gilton et al., 2021] adresses the problem of model adaptation that is tighlty related to DA but more specific.

1.4 Contributions

- We provide a unified view of the domain adaptation problem. We try to better identify the different types of shift that can be encountered. We specially instantiate this for inverse problems.

- Motivated by the impossibility theorems, we further investigate the idea of impossible domain adaptation. Namely, we try to better understand the conditions under which they occur. By doing, we draw a link between the information bottleneck principle and domain adaptation. Finally, we propose an accessible metric to measure the feasibility of DA.

- The methods taken into account in the benchmark are presented in a unified way. As already said, DA literature disposes of plenty for classification of methods. Here, we prefer to unify them to identify their common weaknesses and strengths. And then, we can focus on their differences in the benchmark section.

- We perform numerical experiments on controllable shifts with fair comparison of re-implemented methods. Unlike most of the literature, we will generate our own datasets and introduce the shift arbitrarily. This will allow us to observe the behavior of the methods in a controlled environment to better understand the impact of the shift. Also, as we implemented the methods by ourselves, their architecture can be chosen for the comparison to be fair. The examples on which we will test

the methods are chosen to manifest qualitative behaviors discussed theoretically in the previous sections. At the same time, we create two theoretical impossible cases of DA and observe the results under these conditions¨+.

2 Preliminary notions

To better understand the rest of this work and to present our approach to fundamental aspects, we will introduce some preliminary notions.

2.1 Machine and Deep Learning

Here, we will provide a very brief overview of the concepts of machine learning and deep learning that will be of use in the rest of this work. As explained before, we will be interested in the concept of supervised ML. The theoretical aspects of ML constitute the field of *statistical learning theory* [Vapnik, 2000].

2.1.1 Supervised machine learning

In that setting, we observe a given number of learning pairs, *i. e.* pairs of input and output of an unknown, possibly stochastic mapping. The goal is to obtain an approximation of the mapping to predict outputs on unseen inputs. The step consisting of finding the best approximation leveraging the available pairs is called training and the data, *training set*. Then, at inference time, we use the model on new data points constituting the *test set*. Keeping the problem of DS for later, we make the fundamental assumption that training and test data are sampled from a unique distribution. The distribution of inputs and outputs is denoted $\mathcal{P}_X$ and $\mathcal{P}_Y$ respectively.

More precisely, we will search a given *hypothesis space* $\mathcal{H}$ for the best possible approximation of the mapping. To find the latter, we design a criterion for computing the difference between the predictions of the model and the true outputs from the training set. This is called the *loss function* and is denoted ℓ.

One way to obtain a good hypothesis is to seek the *risk minimizer*. The risk, $R_\ell(h)$ is defined as the expected loss due to a given hypothesis.

$$h^\dagger = \arg\min_{h\in\mathcal{H}} R_\ell(h) \text{ with } R_\ell(h) = \mathbb{E}_{\mathcal{P}}\ell(h(x), y)$$

Nevertheless, the risk minimizer is not accessible in practice. We do not have direct access to the joint distribution $\mathcal{P}_{X,Y}$. The best we can do is constructing an empirical distribution, $\hat{\mathcal{P}}_{X,Y}$ thanks to the training set, $\hat{\mathcal{P}}_{X,Y} = \sum_{i=1}^{N} \delta_{x_i,y_i}$. We finally arrive at the key notion of an empirical risk minimizer (ERM). It is the equivalent of the risk minimizer but based on the empirical distribution.

$$h^* = \arg\min_{h\in\mathcal{H}} \hat{R}_\ell(h) \text{ with } \hat{R}_\ell(h) = \mathbb{E}_{\hat{\mathcal{P}}}\ell(h(x), y) = \frac{1}{N}\sum_{i=1}^{N} \ell(h(x_i), y_i)$$

Nowadays, many hypothesis classes are made of *parametric* models, which means that each hypothesis is defined by a set of parameters $\theta \in \Theta$. The new optimization objective becomes the following.

$$\theta^* = \arg\min_{\theta \in \Theta} \hat{R}_\ell(h_\theta) = \arg\min_{\theta \in \Theta} \frac{1}{N} \sum_{i=1}^{N} \ell(h_\theta(x_i), y_i)$$

Many hypotheses also require tuning hyperparameters that are not learned during the optimization process. To do so, we can extract a validation set from the training set. The training is done on the training set without the validation set. And then, the hyperparamters are chosen to minimize the loss on the validation set. This is called *cross-validation*. By doing this, the model is completely tuned independently from the test set.

One of the most important pitfalls of supervised ML is the risk of *overfitting*. There are many ways to interpret it but we can simply say that it describes the situation where the model performs very well on the training set but poorly on the test set. The usual suspects provoking this are too complex models (too many parameters) or too few training data points.

2.1.2 Neural networks

Neural networks (NN) [Goodfellow et al., 2016] consists of specific types of hypotheses. To make it short and in a very general way, a NN is made of a sequence of non-linear transformations called layers. The current success of NN can be summarized by their great expressivity and flexibility [Bartlett et al., 2021]. A NN disposes of an arbitrarily large number of parameters due to the chosen sequence of layers. Above all, it appears that the NN architecture suffers very little from overfitting regarding its number of parameters, this is called implicit regularization [Bartlett et al., 2021].

The two most important types of layers for the rest of this work are the fully connected layers and the convolutional layers. The first one consists of a matrix multiplication followed by a non-linear transformation. As its name suggests, the convolutional layer is based on the convolution operation, which is also followed by an activation function to introduce nonlinearities. A given representation can be convoluted by an arbitrary number of filters, which corresponds to the number of channels. The way the convolution is handled near the border of the representation is also a choice left to the user. These two hyperparametrizations set the size of the output of the considered layer. The convolutional layer is particularly well suited for visual content due to its translation equivariance. Convolutional Neural Networks (CNN) refers to NN where the majority of layers (or at least the first ones) are convolutional ones.

As explained above in the section on supervised ML, the model is chosen to minimize the empirical risk. In the case of NN, the model is parametric where the parameters are the entries of matrices for fully-connected layer and the convolution kernels for convolutional layers. On top of this, the layers generally add a bias term. The optimization problem is then solved by stochastic gradient descent (SGD) due to the high dimensional aspect and its non-convexity. The gradient is computed through a procedure called backpropagation based on the derivation chain rule.

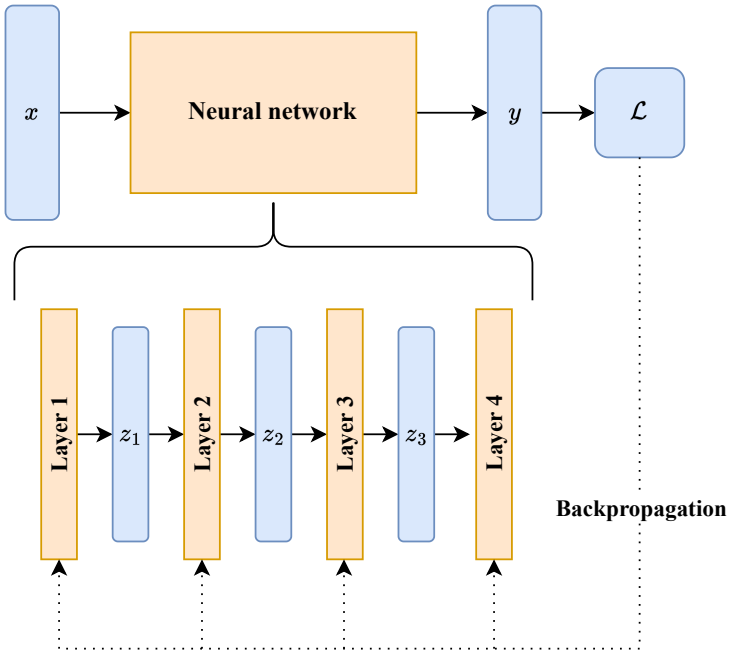


Figure 3: Architecture of a four-layer neural network.

As the dataset may possibly be as large as the number of parameters, the training set is partitioned into *minibatches*. Before updating the parameters, the gradient is computed on the minibatch instead of the whole dataset. The choice of the minibatch at each iteration is random.

We will now present three widely used types of NN that will be of use (at least partially) in this thesis.

AutoEncoder (AE) If this short summary is mainly about supervised machine learning, AE belongs to the realm of unsupervised learning. To make a long story short, in that setting, the model does not try to recover an unknown mapping by observing its input and output, but it tries to discover constitutive patterns in the data. With AE, the user embeds its data in a lower-dimensional space, *the latent space*. To do that, the data point enters a first NN with an output size smaller than its input size. Then, the output vector of the encoder is fed into a second NN, restoring the original shape of the data. A metric measuring the difference between the original input and final output is used as loss indicator. This is represented in figure 4.

The whole idea is to find a representation of the data point that is lower-dimensional and contains the most information about it. Indeed, it was first introduced as a non-linear extension of the Principal Component Analysis (PCA)[Japkowicz et al., 2000].

Convolutional Neural Network (CNN) for classification/regression As we said before, CNN are particularly well suited for visual content. When one wants to classify or predict a continuous value from an image, fully connected layers need to be added at the end of the network. Between the convolutional layers, the user can also add a pooling operation to reduce the size of the representation, introduce translational invariance and

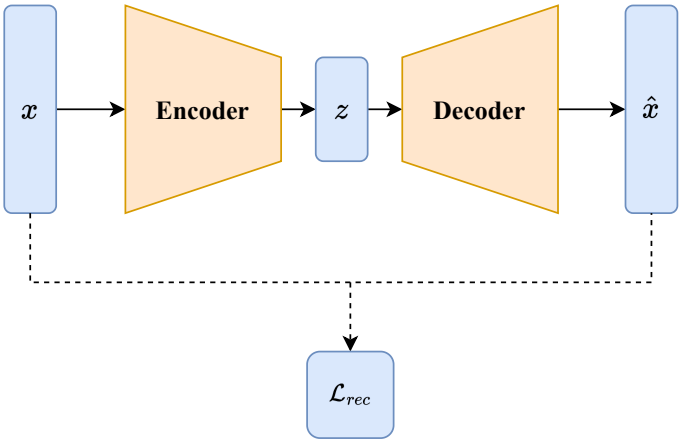


Figure 4: Architecture of an AutoEncoder.

enforce the network to extract the most informative features. The architecture of a CNN is presented in figure 5.

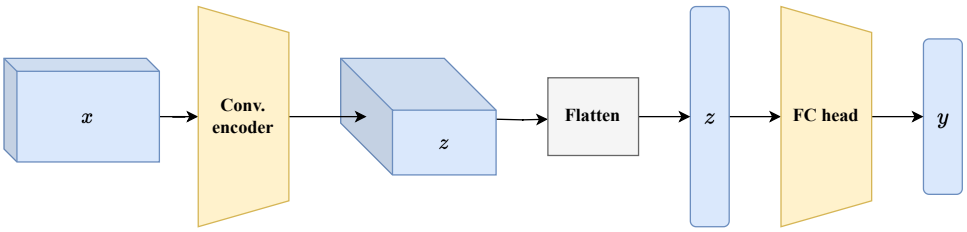


Figure 5: Architecture of a CNN.

U-Net for computational imaging Later in this work, we will consider imaging tasks. Details will come later, but we here present the famous U-Net architecture [Ronneberger et al., 2015]. This architecture has been designed for segmentation tasks, but it can be leveraged for other tasks. Its main interest is its ability to capture local and global features by decomposing the input image into a sequence of representations that are later combined to obtain the final output.

Similarly to the AE, the U-Net is composed of an encoder and a decoder. Each of these blocks is split into different levels. Each level is composed of convolutional layers, activation functions, and a down-or-upsampling operation, respectively, for the encoder and the decoder. The main difference with AE is the skip connections. At each level, the output of the encoder will, on th one hand, continue its encoding at lower levels, on the other hand, it will be forwarded directly to the decoder. The decoder level concatenates both representations (One from the encoder, one from the lower decoder). These skip connections conserve fine details and induce the multi-resolution aspect of U-Net. The architecture is presented on figure 6.

2.1.3 Machine learning and inverse problems

Inverse problems (IP) constitute an important research field for applied mathematicians and engineers. Briefly, it consists in reconstructing an unknown quantity (an image, a

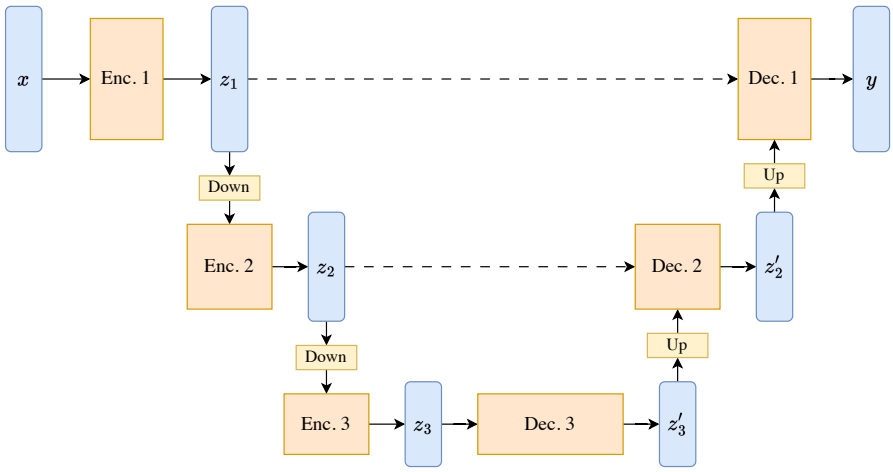


Figure 6: Architecture of a U-Net.

physical field, a sound, *etc.*) from indirect measurements. Most of the time, the observation operator will be assumed to be linear. That yields the following formulation.

$$x = Ay + \eta$$

To remain consistent with the other section, we swap the notations for x and y. One can also consider the mapping to be stochastic and this is embodied by the additive noise η. For the problem to be challenging, the operator A is often ill-posed. This means that the operator is not invertible and that several hidden states may yield the same observation.

Given a new x, when proposing a possible reconstruction $\hat{y}$ of the hidden state, one should verify two criteria. Firstly, the mapped reconstruction should be close to the observation, this is called data consistency. As the operator is not invertible, there are many possible reconstructions that are data-consistent. To fix this, one should ensure that the reconstruction belongs to the prior distribution, *i.e.* the marginal distributions of the hidden states, this is the prior regularization.

More and more, deep learning is being leveraged to solve inverse problems. The neural network should be able to learn the inverse mapping and apply prior regularization implicitly. There exists a rich literature on this, and we will not go into details. In this work, we will consider the specific setting where we dispose of learning pairs generated by an unknown forward operator.

2.2 Information theory for Deep Learning and Information Bottleneck

A major piece of theory in the signal processing domain is information theory. Roughly speaking, it aims at quantifying the entropy of a random variable and how possible it is to predict a variable given another. Lately, this theory has been used in the context of deep learning to understand the behavior of neural networks through the Information Bottleneck (IB) framework [Tishby and Zaslavsky, 2015, Shwartz-Ziv and Tishby, 2017,

Kawaguchi et al., 2023]. Indeed, in forward neural networks, one can understand the output of each layer as a random variable, and it would be interesting to understand the information they share with each other. The IB principle more precisely addresses the question of generalization of deep learning models which is fundamental in DA.

2.2.1 Pieces of information theory

Let us start by recalling the basic definitions of information theory. The fundamental quantity in this realm is entropy. It is a measure of the uncertainty of a random variable. The entropy of a random variable X is defined as follows.

Definition 2.1. *(Entropy)*
Let X be a random variable with a distribution density S. The entropy of X is defined as follows.

$$H(X) = -\int_{x\in\mathcal{X}} S(x)\log S(x)dx$$

where $\mathcal{X}$ is the support of X.

The latter is maximized for uniform distributions and minimized for deterministic ones. Then we can extend it to its conditional variant. Conditional entropy is a measure of the uncertainty of a random variable given another. It is defined as follows.

Definition 2.2. *(Conditional entropy)*
Let X and Y be two random variables. The conditional entropy of X given Y is defined as follows.

$$H(X|Y) = -\int_{x\in\mathcal{X},y\in\mathcal{Y}} S(x,y)\log\frac{S(x,y)}{S(y)}dxdy$$

where $S(x,y)$ is the joint distribution of X and Y and $S(y)$ is the marginal distribution of Y.

If X and Y are independent ($S(x,y) = S(x)S(y)$), then $H(X|Y) = H(X)$. The conditional entropy is a measure of the uncertainty of X given Y. If X is a deterministic function of Y, then $H(X|Y) = 0$.

Following these two extreme cases, we realize that one can measure the dependency between two random variables by the difference between their marginal and conditioned entropy. This quantity is called mutual information.

Definition 2.3. *(Mutual information)*
Let X and Y be two random variables. The mutual information between X and Y is defined as follows.

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X)$$

The mutual information is a measure of the dependency between X and Y.

As for entropy, we can extend the notion of mutual entropy to the conditional case. If Z is a third random variable, the conditional mutual information $I(X;Y|Z)$ quantifies the dependency between X and Y when Z is fixed.

Definition 2.4. *(Conditional mutual information)*
Let X, Y and Z be three random variables. The conditional mutual information between X and Y given Z is defined as follows.

$$I(X;Y|Z) = H(X|Z) - H(X|Y,Z)$$

The conditional mutual information is a measure of the dependency between X and Y when Z is fixed.

A useful equality involving mutual information is the chain rule of mutual information. It states that the mutual information between two random variables can be decomposed into the sum of the mutual information between one of the random variables and a third one, and the mutual information between the two random variables given the third one.

$$I(X;Y,Z) = I(X;Z) + I(X;Y|Z) \tag{1}$$

Finally, we introduced two more information-theoretic properties. The first one tells us that conditioning can only reduce the entropy of a random variable.

$$H(X|Y) \leq H(X) \text{ for any random variable } Y \tag{2}$$

The second one is the data processing inequality. It states that for any Markov chain $X \to Z \to Y$, the intermediate variable Z cannot possess more information about Y than X.

$$I(X;Y) \geq I(X;Z) \text{ for any Markov chain } X \to Z \to Y \tag{3}$$

2.2.2 Information Bottleneck

In its seminal work [Tishby et al., 2000], IB was introduced as a method of supervised ML with high generalization abilities. The main idea is to discover a latent representation Z of the input X that is informative about the output Y. The latent code must be as informative as possible about Y while being as independent as possible from X. Without going into details, it maximizes the following objective.

$$\max_{p(z|x)} I(Z;Y) - \beta I(Z;X) \tag{4}$$

Once the transition probability $p(z|x)$ is found, the model can be trained on the latent representation Z instead of the input X. While the first term of the objective is obvious, the second one is more subtle. It promotes generalization by forcing the latent representation to be as independent as possible from the input. Therefore, we hope that Z is not influenced by irrelevant features of X.

Besides the objective 4, several other metrics have been introduced to measure the quality of the latent representation [Kirsch et al., 2021].

Name	Expression	Interpretation
Relevant Information	$I(X;Y)$	Quantity of information that a model can capture to predict Y from X. This quantity depends only on the dataset and cannot be influenced.
Preserved Information	$I(X;Z)$	Quantity of information on X still available in Z.
Preserved Relevant Information	$I(Y;Z)$	Quantity of information that a model can exploit to predict Y. The IB objective is to maintain this metric while minimizing the preserved information.
Residual Information	$I(X;Y\|Z)$	Quantity of information that X contains about Y that is not available in Z. In a sense, it measures the quality of the encoder : if it is positive, this means that there still exist informative features in X that are missed by the code Z.
Redundant Information	$I(X;Z\|Y)$	Quantity of information that Z contains about X that is not needed to predict Y.

Table 1: Metrics used in the IB framework. Gathered in [Kirsch et al., 2021].

From the chain rule 1, we can derive the following about relevant and preserved information.

$$I(X;Y) = \underbrace{I(X;Y|Z)}_{\text{Residual}} + I(Y;Z)$$
$$I(X;Z) = \underbrace{I(X;Z|Y)}_{\text{Redundant}} + I(Y;Z)$$

This reformulation emphasizes that maximizing 4 is equivalent to the subtle balance between redundancy and residual information.

The IB framework has been naturally extended to deep learning. The objective is not anymore to directly maximize the cost function from 4 but to observe how these quantities evolve along layers. This is a relevant tool to understand the behavior of NN by observing how these information-theoretic metrics are influenced by the architecture [Shwartz-Ziv and Tishby, 2017, Achille and Soatto, 2018]. In [Kirsch et al., 2021], it has been shown that classical deep learning strategies (classical loss backpropagation, dropout layers, *etc.*) are good surrogates for the IB objectives.

The notion of latent code was already introduced when explaining the autoencoder architecture. Here, we consider the outputs of each layer as latent code.

2.3 Statistical distances and optimal transport for domain adaptation

Through the very general idea of distribution shift, the question of measuring the difference between two distributions arises. Writing that two distributions are different $\mathcal{S} \neq \mathcal{T}$ is never informative. Indeed, two distributions of data with a slight class-imbalance are different, as are two distributions of data with different modalities.

Therefore, statistical distances are supposed to provide an interesting measure of the differences between distributions. Consequently, these distances will be paramount in this work. Theoretical analysis of domain adaptation methods will be based on these distances. And also, computing them will be a key part of the DA methods.

This subsection aims at presenting the distances needed and their important related notions for the rest of this work. For the sake of simplicity, we always equivalently use the terms distance, divergences, or discrepancy to encompass all the metrics, without considering the subtle differences that are sometimes present between them.

2.3.1 Classical distances

The applied mathematics connoisseur has probably already thought of some metrics when citing statistical distances. Some of them are very general and find applications in many fields, including DA. We will shortly recall them here.

Total Variation (TV) distance : Probably the most common distance between distributions. It is defined as $TV(\mathcal{S}, \mathcal{T}) = \sup_{\xi \subseteq X} |\mathcal{S}(\xi) - \mathcal{T}(\xi)|$. A limiting aspect of this metric regarding our application is its tendency to saturate when supports do not perfectly overlap. For example, the distance between a uniform distribution $\mathcal{S} = \text{Unif}(0, 1)$ and an empirical one $\hat{\mathcal{S}} = \frac{1}{n} \sum_{i=1}^{n} \delta_{x_i}$ with $x_i \sim \mathcal{S}$ is $TV(\mathcal{S}, \hat{\mathcal{S}}) = 1$ for any n which is the maximum value TV can reach.

Kullback-Leibler (KL) and Jensen-Shannon (JS) divergence : This is also a well-known statistical distance. As its alternative name, relative entropy, indicates, KL divergence has been developed in the context of information theory. Its definition is :

$$KL(\mathcal{S}||\mathcal{T}) = \int_X x \log\left(\frac{S(x)}{T(x)}\right) S(x)dx$$

It represents the expected amount of surprise when using the distribution $\mathcal{Q}$ to approximate the distribution $\mathcal{P}$. The latter is not symmetric. Consequently, the JS divergence has been introduced as the symmetrized version of the KL divergence. It is defined as follows :

$$JS(\mathcal{S}, \mathcal{T}) = \frac{1}{2} KL(\mathcal{S}||\mathcal{M}) + \frac{1}{2} KL(\mathcal{T}||\mathcal{M}) \text{ with } \mathcal{M} = \frac{1}{2}(\mathcal{S} + \mathcal{T})$$

Regarding information theory, we can find an intuitive interpretation. It considers an even mixture of the distributions $\mathcal{S}$ and $\mathcal{T}$ with the mixture indicator $\mathcal{N} = \text{Ber}(1/2)$. The

JS divergence represents the mutual information between one of the distributions and the mixture distribution.

$$JS(\mathcal{S}, \mathcal{T}) = I(\mathcal{S}; \mathcal{N}) = I(\mathcal{T}; \mathcal{N})$$

As is often the case with information-theoretic quantities, despite their convenient interpretation, they are not always easy to apply to continuous variables. Here, we need strong assumptions on the domains of the distributions for the metrics to be defined and meaningful. The KL divergence will not be defined if the support of $\mathcal{T}$ is not included in the support of $\mathcal{S}$. The JS divergence will be maximal (one bit) if the supports of the two distributions do not overlap. Yet, they have found applications in recent ML advances. On the one hand, with the advent of variational approaches, it has been possible to involve these divergences in deep learning. For example, it is fundamental in the Variational AutoEncoder (VAE) [Kingma and Welling, 2022]. Also, the GAN adversarial objective can be seen as a proxy for the JS divergence [Goodfellow et al., 2014]. Both of these metrics and their link with deep learning will be of interest in the rest of this work.

On top of these classical distances that have been developed in numerous contexts, some distances have been more or less directly designed for DA. The most important are the following.

2.3.2 Maximum Mean Discrepancy

This metric has not been directly designed for DA. Yet, it has found an important place in this field. Its seminal idea comes from a statistical test, the two-sample problem [Gretton et al., 2012]. The latter consists of evaluating the hypothesis that two samples are issued from the same distribution or not. MMD will be directly involved in domain adaptation methods through empirical estimates. This section aims at providing definitions and motivations for this metric, on the one hand. On the other hand, we will present the empirical estimates that are directly implemented in the methods. Finally, we will present its multi-kernel extension, which will also be of prime interest for the following work. All the theoretical developments about the consistency of the estimates and their relevance regarding the two-sample problem will be omitted. The reader can refer to [Gretton et al., 2008, Gretton et al., 2012] for more details.

To understand the relevance of the Maximum Mean Discrepancy (MMD), let's first recall a general theorem about measurable spaces from[Müller, 1997]. The latter tells us the following.

Theorem 2.1. *(Proof in [Müller, 1997])*
Let $\mathcal{S}$ be a probability measure on X. Let $\mathcal{T}$ be another measure on the same space. Then, $\mathcal{S} = \mathcal{T}$ if and only if for all f continuous and differentiable:

$$\int_X f(x)S(x)dx = \int_X f(x)T(x)dx$$

In the case of probability densities, the condition becomes:

$$\mathbb{E}_{x\sim\mathcal{S}} f(x) = \mathbb{E}_{x\sim\mathcal{T}} f(x)$$

For the first time, the notion of Integral Probability Metric (IPM) has been introduced [Müller, 1997].

Definition 2.5. *(Integral Probability Metric)*
Let $\mathscr{F}$ be a class of differentiable and continuous functions. The IPM is defined as follows.

$$D_{\mathscr{F}}(\mathcal{S}, \mathcal{T}) = \sup_{f \in \mathscr{F}} |\mathbb{E}_{x \sim \mathcal{S}} f(x) - \mathbb{E}_{x \sim \mathcal{T}} f(x)|$$

the function f is usually called the witness or critic function.

From theorem 2.1, we directly deduce that $D_{\mathscr{F}}(\mathcal{S}, \mathcal{T}) = 0$ if and only if $\mathcal{S} = \mathcal{T}$.

Now, one must be careful : when introducing the MMD, the authors described it as being the exact same thing as the IPM. The term MMD has been more widely used since then. By metonymy, the MMD also refers to the way it is approximated in practice. Some authors (for example [Redko et al., 2019, Long et al., 2015]) name MMD the restriction to function classes made of unit balls of a Reproducing Kernel Hilbert Space (RKHS) $\mathscr{H}$, and so will we do for the rest of this work.

Definition 2.6. *(Maximum Mean Discrepancy)*
Under the same assumptions as definition 2.5, the MMD is defined as follows.

$$MMD(\mathscr{F}; \mathcal{S}, \mathcal{T}) = \sup_{f \in \mathscr{F}} |\mathbb{E}_{\mathcal{S}} f(x) - \mathbb{E}_{\mathcal{T}} f(x)|$$

Now, we will introduce a much more useful version of the MMD : its empirical estimate. For that, we first consider its application to a RKHS $\mathscr{H}$. Thanks to the definition of norm in a Hilbert space, we obtain the following lemma.

Kernel operator in Reproducing Kernel Hilbert Spaces For the sake of shortness, we will not go into a detailed explanation of RKHS. We will only loosely recall the notions that will be of use for the kernel MMD to be understood.

When treating data, the underlying notions of inner product or norm are of prime importance. For example, one may be interested in computing the distance between two instances or how correlated two features are. Although classical Euclidean distance or inner product may not be particularly adapted to the data, to tackle this, one could first extract new features through a feature map $\phi(\cdot)$ from the data and then perform the desired computation. RKHS let us combine these two steps in a single one. Its kernel $k(\cdot, \cdot)$ is a function that return the inner product of the feature map of two instances, $k(x, y) = \langle \phi(x), \phi(y) \rangle$. To finally define properly the RKHS, the kernel has to verify the so-called reproducing property. We will not go into details about this property as it is not necessary for the understanding of the MMD as we present it. The meticulous reader may refer to [Wahba, 2003] to learn more about this.

Lemma 2.2. *(Proof in [Gretton et al., 2008], lemma 4)*
Let $\mathscr{H}$ be a RKHS. Let k be the reproducing kernel of $\mathscr{H}$. Then, the MMD can be written as follows.

$$MMD(\mathscr{H}; \mathcal{S}, \mathcal{T}) = ||\mathbb{E}_{\mathcal{S}} \phi(x) - \mathbb{E}_{\mathcal{T}} \phi(x)||_{\mathscr{H}}$$

where ϕ is the feature map of $\mathscr{H}$.

Sometimes $MMD(\mathscr{H};\mathcal{S},\mathcal{T})$ is denoted directly with the feature map $MMD(\phi;\mathcal{S},\mathcal{T})$ to generalize out of its initial kernel framework. Based on the latter result and developing the norm in order to make the kernel of $\mathscr{H}$ appear $k(\cdot,\cdot)$, we obtain the following expression.

Lemma 2.3. *(Proof in [Gretton et al., 2008], lemma 5)*
Let $\mathscr{H}$ be a RKHS. Let k be the reproducing kernel of $\mathscr{H}$. Then, the MMD can be written as follows.

$$MMD^2(\mathscr{H};\mathcal{S},\mathcal{T}) = \mathbb{E}_{x,x'\sim\mathcal{S}}[k(x,x')] + \mathbb{E}_{y,y'\sim\mathcal{T}}[k(y,y')] - 2\mathbb{E}_{x,y\sim\mathcal{S},\mathcal{T}}[k(x,y)]$$

When the context is clear enough, we will directly write $MMD(k;\mathcal{S},\mathcal{T})$ instead of $MMD(\mathscr{H};\mathcal{S},\mathcal{T})$ to detach from the RKHS framework that will not be fundamental in our numerical experiments. From this final lemma, we can derive the empirical estimates of the MMD. These versions are the ones that will be of interest in a practical setting.

Definition 2.7. *(Empirical biased MMD [Gretton et al., 2008])*
Let $\mathscr{H}$ be a RKHS. Let $k(\cdot,\cdot)$ be the reproducing kernel of $\mathscr{H}$. Let $\mathfrak{S} = \{x_1^S,\dots,x_m^S\}$ and $\mathfrak{T} = \{x_1^T,\dots,x_n^T\}$ be two samples. The empirical MMD is defined as follows.

$$MMD_B(\mathscr{H};\mathfrak{S},\mathfrak{T}) = \sqrt{\frac{1}{m^2}\sum_{i,j=1}^{m} k(x_i^S,x_j^S) + \frac{1}{n^2}\sum_{i,j=1}^{n} k(x_i^T,x_j^T) - \frac{2}{mn}\sum_{i,j=1}^{m,n} k(x_i^S,x_j^T)}$$

The first expression (definition 2.7) is quite intuitive as it directly substitutes the expectations by empirical averages. However, this expression is biased. But it is possible to obtain an unbiased estimate.

Definition 2.8. *(Empirical unbiased MMD [Gretton et al., 2008])*
Let $\mathscr{H}$ be a RKHS. Let $k(\cdot,\cdot)$ be the reproducing kernel of $\mathscr{H}$. Let $\mathfrak{S} = \{x_1^S,\dots,x_m^S\}$ and $\mathfrak{T} = \{x_1^S,\dots,x_m^T\}$ be two samples. The empirical MMD is defined as follows.

$$MMD_U(\mathscr{H};\mathfrak{S},\mathfrak{T}) = \sqrt{\frac{1}{m(m-1)}\sum_{i\neq j} h(x_i^S,x_j^S;x_i^T,x_j^T)}$$

where we define $h(x_i^S,x_j^S;x_i^T,x_j^T) = k(x_i^S,x_j^S) + k(x_i^T,x_j^T) - k(x_i^S,x_j^T) - k(x_j^S,x_i^T)$.

Multi-kernel MMD Finally, the notion of MMD finds its final extension in the concept of multi-kernel MMD (MK-MMD). This discrepancy metric was introduced in [Gretton et al., 2012]. To understand its motivation, let us go back to the statistical test that has been the starting point of the MMD. The latter aims at deciding if two samples are coming from the same distribution. MK-MMD was created to reduce the risk of type II error in this test, *i.e.* postulating that the samples are coming from the same distribution while they are not. This motivation fits well with the practical objectives involved in DA as it will be detailed later.

Now, regarding the MK-MMD in itself, the key idea is to tune the reproducing kernel to be more sensitive to discrepancies. The kernel k is chosen as a convex combination of n_K different kernels k_p spanning a kernel set $\mathscr{K}$.

$$k(\cdot,\cdot) \in \mathscr{K} = \Big\{ \sum_{p=1}^{n_K} \beta_p k_p(\cdot,\cdot) \Big| \sum_{p=1}^{n_K} \beta_p = 1 \text{ and } \beta_p \geq 0 \Big\} \tag{5}$$

In practice, these kernels are issued from the same family of kernels but with different hyperparameters. With MK-MMD, an optimization problem has to be solved to find the mixing coefficients β_p. The latter consists in a quadratic constrained program with n_K variables, so it can be solved efficiently.

Following our philosophy to skip some parts of the how and why of the underlying theory in this review of statistical metrics, we will not go into the details of the justification of the optimization problem (that is thoroughly explained in [Gretton et al., 2012]). The reasoning is based on a minimization of the variance of $h(\cdot,\cdot;\cdot,\cdot)$ (defined in definition 2.8). The expression of the optimization problem is the following.

$$\max_{k \in \mathscr{K}} \frac{MMD^2(k, \mathcal{S}, \mathcal{T})}{\sigma_k^2} \tag{6}$$

where σ_k^2 is the variance of the $h(\cdot,\cdot;\cdot,\cdot)$ under the kernel k. And as an easy-to-use version, we can inject the specific structure of $\mathscr{K}$ (equation 5) in the optimization problem to obtain the following.

$$\min_{\beta} \beta^\top \mathbf{Q} \beta \text{ subject to } \beta^\top \mathbf{1} = 1, \beta \geq 0 \tag{7}$$

where $\mathbf{Q} = (\mathbf{cov}(h(x_i^S, x_j^S; x_i^T, x_j^T)) + \epsilon \mathbf{I})$ is the covariance matrix (with ϵ a small constant that ensures the positive-definiteness of the $\mathbf{Q}$.

2.3.3 Optimal transport

The next distance of interest is the Wasserstein distance. But before, we need to introduce the closely related optimal transport (OT) problem. By doing this, we kill two birds with one stone, as optimal transport in itself will also be of help later in this work.

Also known as the Monge-Kantorovich problem, it aims at finding the most efficient way to transport a distribution of mass to another. Let's consider the two distributions $\mathcal{S}$ and $\mathcal{T}$ on a metric space (X, d). They should not especially be probability distributions. If one wants to displace the mass from $\mathcal{S}$ to $\mathcal{T}$, the cost of moving a mass element from x to y is given by the distance $d(x, y)$. The optimal transport problem consists in finding the transport plan γ that minimizes the total cost of the transport. The optimal cost is the Wasserstein distance. We can formalize this as follows.

Definition 2.9. *(Optimal transport problem)*
Let $\mathcal{S}$, $\mathcal{T}$ be two distributions on a metric space of interest. The optimal transport problem is defined as follows.

$$\gamma = \arg\min \left(\int_{X \times X} d(x_S, x_T)^p d\gamma(x_S, x_T) \right)^{\frac{1}{p}}$$

with γ verifying :

$$\int_{X_S} \gamma(x_S, x_T) S(x_S) dx = T(x_T) \text{ and } \int_{X_T} \gamma(x_S, x_T) T(x_T) dy = S(x_S)$$

One of the underlying ideas of its application to DA is the following: Let us suppose that there exists a push-forward mapping between the two distributions of interest.

$$\exists \gamma^{\dagger} : X \to X \text{ s.t. } \gamma^{\dagger}_{\#}\mathcal{S} = \mathcal{T} \Longleftrightarrow \mathcal{T}(x_T) = \mathcal{S}(\gamma^{-\dagger}(x_T))$$

where $\gamma^{-\dagger}$ is the inverse of $\gamma^{\dagger}$. Of course, there may exist an unbounded number of push forward mappings between two distributions. The OT problem tries to find the one that minimizes a given cost function.

To better understand it, let's consider a case that will also be very useful in practical implementations : empirical distributions [Courty et al., 2017b]. Let $\mathfrak{S} = \{x_1^S, \dots, x_m^S\}$ and $\mathfrak{T} = \{x_1^T, \dots, x_n^T\}$ be two samples. The empirical distributions are defined as $\hat{\mathcal{S}} = \frac{1}{m}\sum_{i=1}^{m} \delta_{x_i^S}$ and $\hat{\mathcal{T}} = \frac{1}{n}\sum_{i=1}^{n} \delta_{x_i^T}$. Let's denote Γ the set of acceptable transport plans.

$$\Gamma = \{\gamma \in \mathbb{R}_+^{m \times n} | \gamma \geq 0, \gamma \mathbf{1}_n = \frac{1}{m}\mathbf{1}_m, \gamma^T \mathbf{1}_m = \frac{1}{n}\mathbf{1}_n\}$$

Also, we need to construct the pairwise distance matrix C such that $C_{ij} = ||x_i^S - x_j^T||$. The optimal transport problem becomes:

$$\gamma = \arg\min_{\gamma \in \Gamma} \langle \gamma, C \rangle_F$$

In the case where both datasets have the same size, it corresponds exactly to the optimal assignment problem.

2.3.4 Wasserstein distance

Regarding the optimal transport problem exposed here above, the definition of the p-Wasserstein distance is directly deduced. We look for a transport plan $\gamma \in \Gamma$ that minimizes the total cost of the transport. The set Γ embodies the constraint that all the initial mass has to be transported to the final one.

Definition 2.10. *(Wasserstein distance)*
Let $\mathcal{S}$, $\mathcal{T}$ be two distributions on a metric space of interest, and $\Gamma(\mathcal{S}, \mathcal{T}) = \{\gamma | \int_{X_S} \gamma(x_S, x_T) S(x_S) dx_S = T(x_T)$ and $\int_{X_T} \gamma(x_S, x_T) T(x_T) dx_T = S(x_S)\}$. The p-Wasserstein distance is given by :

$$W_p(\mathcal{S}, \mathcal{T}) = \inf_{\gamma \in \Gamma(\mathcal{S}, \mathcal{T})} \left(\int_{X \times X} d(x_S, x_T)^p d\gamma(x_S, x_T) \right)^{\frac{1}{p}}$$

Hereafter, we will mainly consider the Wasserstein 1-distance, referred to just as the Wasserstein distance and denoted $W(\cdot, \cdot)$. Also, a very useful result about that metric will be of help: the Kantorovich-Rubinstein (KR) duality. The latter provides us with an alternative definition and intuition about the metric.

Theorem 2.4. *(Kantorovich-Rubinstein duality) [Edwards, 2011]*
Let $\mathcal{S}$, $\mathcal{T}$ be two distributions on a metric space (X, d). Let $f : X \to \mathbb{R}$ be a Lipschitz-continuous function. Then, the Wasserstein distance can be defined as follows for all $K > 0$.

$$W_1(\mathcal{S}, \mathcal{T}) = \frac{1}{K} \sup_{||f||_{\mathscr{L}} \leq K} |\mathbb{E}_{x \sim \mathcal{S}} f(x) - \mathbb{E}_{x \sim \mathcal{T}} f(x)|$$

where $||f||_{\mathscr{L}}$ is the Lipschitz constant of f.

Measuring the gap between expectations of two distributions is quite a naive metric, which may not be very informative. The KR duality looks for a *magnifying glass*, f that highlights the difference between $\mathcal{S}$ and $\mathcal{T}$. For the result not to be arbitrarily large, the Lipschitz constant is bounded. One can also directly link the KR duality with definition 2.5. Indeed, one can see the Wasserstein distance as an IPM using the class of Lipschitz-continuous functions.

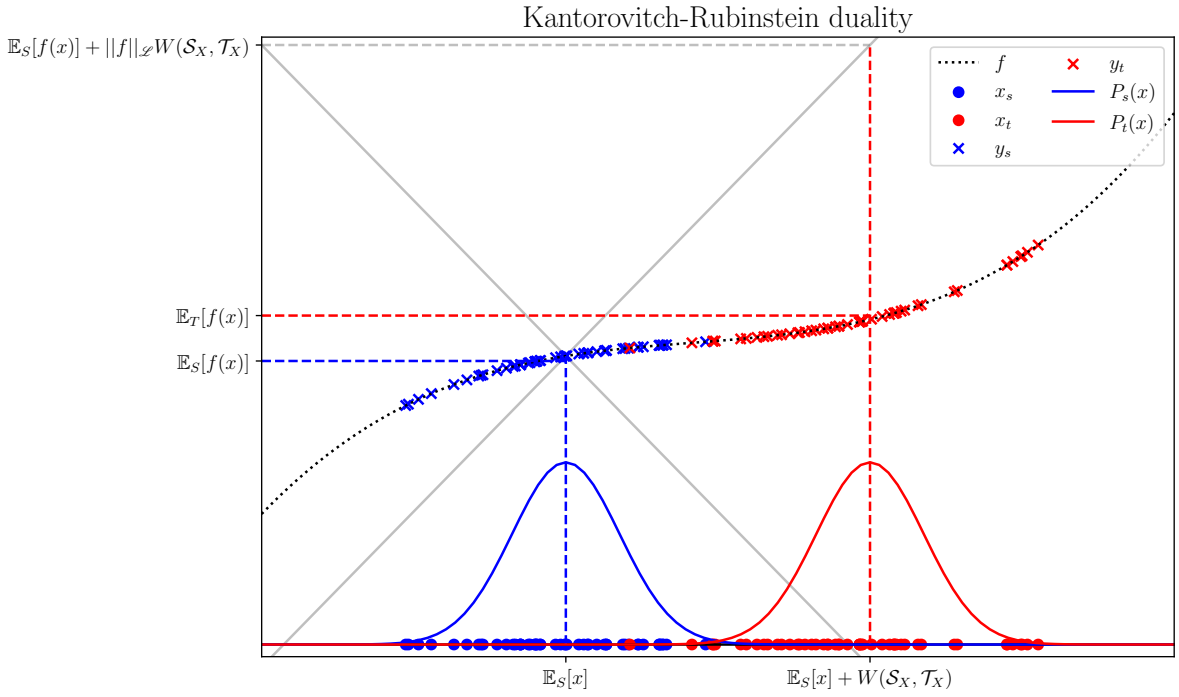


Figure 7: Intuitive idea behind the Kantorovich-Rubinstein duality. Whereas the Lipschitz constant quantifies the maximum gap between output given distance between inputs. Here, we extend this idea to the probabilistic setting. And instead of the distance between inputs, we consider the Wasserstein distance between the distributions.

There are several consequences of the KR duality that will be of use later as lemmas to compute a learning bound. The first one tells us that if two joint distributions $\mathcal{S}, \mathcal{T}$ have the same conditional distributions $\mathcal{S}_{Y|X} = \mathcal{T}_{Y|X}$ (which will be useful when considering covariate shift), and the Wasserstein distance is equal to the distance between the marginals $\mathcal{S}_X = \mathcal{T}_X$.

Lemma 2.5. *Proof in appendix C*
Let $\mathcal{S}, \mathcal{T}$ be two joint distributions on a metric space. If $\mathcal{S}_{Y|X} = \mathcal{T}_{Y|X}$ and the conditional distributions follows the following Wasserstein-Lipschitz K-smoothness condition : $\forall x, x' \in X, W(\mathcal{S}_{Y|X=x}, \mathcal{S}_{Y|X=x'}) \leq K d(x, x)$.

$$W(\mathcal{S}, \mathcal{T}) \leq (1 + K) W(\mathcal{S}_X, \mathcal{T}_X)$$

We can extend the last results to the case of a conditional shift as well.

Lemma 2.6. *Proof in appendix C*
Let $\mathcal{S}, \mathcal{T}$ be two joint distributions on a metric space with conditional distributions that are Wasserstein-Lipschitz K-smooth.

$$W(\mathcal{S}, \mathcal{T}) \leq (1+K)W(\mathcal{S}_X, \mathcal{T}_X) + \min\left\{\mathbb{E}_{S_X}[W(\mathcal{S}_{Y|X}, \mathcal{T}_{Y|X})], \mathbb{E}_{T_X}[W(\mathcal{S}_{Y|X}, \mathcal{T}_{Y|X})]\right\}$$

2.3.5 Other distances for domain adaptation

To conclude our overview of useful distances, a few more divergences are introduced here. They have been specifically designed to express theoretical guarantees under distribution shifts.

$\mathcal{A}$-divergence [Ben-David et al., 2006] : This is a direct restriction on the TV distance. Here, instead of searching the whole space of measurable sets, we restrict ourselves to a smaller set of sets.

Let $\mathcal{A}$ be a class of measurable sets. The $\mathcal{A}$-divergence is defined as follows.

$$D_{\mathcal{A}}(\mathcal{S}, \mathcal{T}) = \sup_{\xi \in \mathcal{A}} |\mathcal{S}(\xi) - \mathcal{T}(\xi)|$$

The main advantage of this reformulation of the TV is that we can introduce consideration of the hypothesis class in the case of binary classification. In the latter situation, the set $\mathcal{A}$ is the set of subsets that can receive the same label under a hypothesis from the class $\mathcal{H}$. In this case, we call it the $\mathcal{H}$-divergence and denote it $D_{\mathcal{H}}(\cdot, \cdot)$.

Symmetric difference hypothesis divergence [Ben-David et al., 2010] : This divergence is the direct extension of the $\mathcal{H}$-divergence ($\mathcal{A}$-divergence where $\mathcal{A}$ is linked to the hypothesis class). Here, the set $\mathcal{A}$ is not directly made of a subset that can receive the same label under a hypothesis from the class $\mathcal{H}$. We would rather consider the symmetric difference hypothesis space $\mathcal{H}\Delta\mathcal{H}$. This set is defined as :

$$g \in \mathcal{H}\Delta\mathcal{H} \Leftrightarrow \exists h_1, h_2 \in \mathcal{H} \text{ s.t. } g(x) = h_1(x) \oplus h_2(x) \tag{8}$$

where $\oplus$ is the XOR operator.

Discrepancy distance [Mansour et al., 2009] : The discrepancy distance is very similar to the previous one but generalizes out of the context of binary classification.

For this divergence, we must define a relative risk between two hypothesis $h_1, h_2 \in \mathcal{H}$ under a density distribution S_X.

$$R_S(h_1, h_2) = \mathbb{E}_{S_X}\left[\ell(h_1(x), h_2(x))\right]$$

That being said, the discrepancy distance is defined as follows.

$$disc(\mathcal{S}, \mathcal{T}) = \sup_{h_1, h_2 \in \mathcal{H}} |R_S(h_1, h_2) - R_T(h_1, h_2)|$$

The three above divergences will be directly involved in the next section to express learning bounds. So will do the following one, but the latter will also introduce a key concept about DA that we will discuss later.

Support sufficiency divergence [Johansson et al., 2019]

Divergences can be more or less sensitive to support overlap. This metric has been designed to derive a new learning bound and to highlight the importance of domain overlap in domain adaptation. Support sufficiency divergence is arbitrarily sensitive to domain discrepancy.

Definition 2.11. *(Support sufficiency divergence)*
Let $\mathcal{S}$, $\mathcal{T}$ be two distributions. The support sufficiency divergence is defined as follows.

$$D^{\epsilon}_{sup}(\mathcal{S}||\mathcal{T}) = \mathbb{E}_T[\varpi^{\epsilon}_{\mathcal{S},\mathcal{T}}(x)] - \mathbb{E}_S[\varpi^{\epsilon}_{\mathcal{S},\mathcal{T}}(x)]$$

with the following definition for $\varpi^{\epsilon}_{\mathcal{S},\mathcal{T}}$.

$$\varpi^{\epsilon}_{\mathcal{S},\mathcal{T}}(x) = \begin{cases} 1 & \text{if } T(x) > S(x) \text{ and } S(x) < \epsilon \\ 0 & \text{otherwise} \end{cases}$$

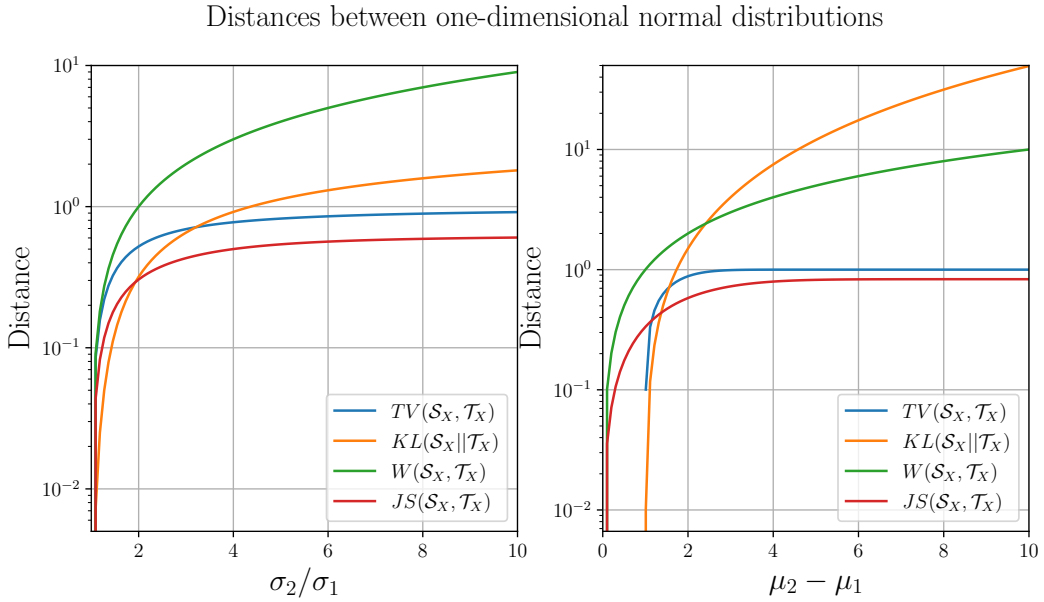


Figure 8: Evolution of different statistical distances between two one-dimensional normal distributions. One can see that Kullback-Leibler explodes rapidly when the two distributions do not overlap. On the other hand, total variation saturates rapidly. In between, the Wasserstein distance is the most stable.

3 Statistical learning theory under distribution shift

Before exploring more concrete aspects of domain adaptation, let us first summarize the main theoretical guarantees we dispose of in the distribution shift scenario. Most of the results apply to specific simple cases (binary classification, one-dimensional regression, deterministic labeling, ...). Despite this, we can informally extend their underlying concepts to better understand the success and failure of domain adaptation. More precisely, we will identify the key aspects of these pieces of theory to better interpret the results.

3.1 Learning bounds

The learning bounds take the form of upper bounds on the target risk. In other words, it tells us what phenomena may lead to decreased performance. They provide sufficient conditions for our learning procedure to succeed. We will first list some bounds that are relevant with respect to the aspects considered in this work. Then, we will briefly discuss a common structure and how tractable it is.

3.1.1 Incremental review of learning bounds

The first bounds that have been derived in the context of DA come from [Ben-David et al., 2006]. It employs the TV distance and the $\mathcal{H}$-divergence.

Theorem 3.1. *(Proof in [Ben-David et al., 2006])*
Let $f_S : \mathcal{X} \to \mathcal{Y}$ and $f_T : \mathcal{X} \to \mathcal{Y}$ the true labeling functions of the source and target domains. Let $h \in \mathcal{H}$ be a hypothesis. Then, the following holds for a binary classification task.

$$R_T(h) \leq R_S(h) + TV(\mathcal{S}, \mathcal{T}) + \min\big\{\mathbb{E}_S||f_S(x) - f_T(x)||, \mathbb{E}_T||f_S(x) - f_T(x)||\big\}$$

This seminal expression already introduces the general form of learning bounds in domain adaptation that we will discuss later to propose a unifying view. But the major drawback of the latter is that it does not take into account the hypothesis class. This matters in DA. Let's imagine a hypothesis class that is totally invariant to the transport map between source and target (for example, rotation). In that setting, the statistical distance between inputs from the source and target is not relevant anymore. For this reason, the next bound has been derived in [Ben-David et al., 2010].

Theorem 3.2. *(Proof in [Ben-David et al., 2010])*
Let $\lambda_{\mathcal{H}} = \min_{h \in \mathcal{H}}[R_S(h) + R_T(h)]$. Let $h \in \mathcal{H}$ be a hypothesis. Then, the following holds for a binary classification task with a deterministic labeling procedure.

$$R_T(h) \leq R_S(h) + d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S}, \mathcal{T}) + \lambda_{\mathcal{H}}$$

The two previous bounds only considered binary classification, we are still far from multi-class or regression tasks. That is why it comes up with a third bound in our incremental review. This one is valid for binary classification and one-dimensional regression ($\mathcal{Y} = \mathbb{R}$). It has been introduced in [Mansour et al., 2009]. This bound also introduces concerns about the ability of the hypothesis class to perform on the target domain by involving the target risk-minimizer, h_T^*.

Theorem 3.3. *(Proof in [Mansour et al., 2009])*
The following holds for a binary classification task or one-dimensional regression with a deterministic labeling procedure.

$$R_T(h) \leq R_T(h_T^*) + R_S(h, h_S^*) + disc(\mathcal{S}, \mathcal{T}) + R_T(h_T^*, h_S^*)$$

Now, we can also extend to non-deterministic labeling procedures (the label y is not a function of x but sampled from a distribution $\mathcal{S}_{Y|X}$ or $\mathcal{T}_{Y|X}$). This assumption does not differ much in the usual cases but will be relevant later when considering latent representation instead of raw inputs.

Theorem 3.4. *(Proof in appendix C)*
Let h be a Lipschitz-continuous hypothesis. If the conditional distributions are Wasserstein-Lipschitz K-smooth, there exists a constant C such that the following holds.

$$R_T(h) \leq R_S(h) + C\Big(W(\mathcal{S}_X, \mathcal{T}_X) + \min\big\{\mathbb{E}_{S_X}[W(\mathcal{S}_{Y|X}, \mathcal{T}_{Y|X})], \mathbb{E}_{T_X}[W(\mathcal{S}_{Y|X}, \mathcal{T}_{Y|X})]\big\}\Big)$$

Where $C = ||\ell||_{\mathscr{L}}(||h||_{\mathscr{L}} + 1)(1 + K)$.

The appearance of the Lipschitz constant in the bound is not problematic, even if we consider neural networks as a hypothesis class. In [Redko et al., 2019], a similar bound was obtained for RKHS, where the Lipschitz constant can be upper bounded for all hypotheses. One cannot do so for neural networks, but one can approximate the Lipschitz constant in a restricted domain.

One can also find a similar bound in [Nguyen et al., 2021a] using Kullback-Leibler divergence instead of the Wasserstein metric. Involving KL divergence justifies methods, but for theoretical analysis, we prefer the Wasserstein distance as it is more robust to support divergence between domains.

Finally, we present here a bound that will bring an important notion to domain adaptation. The latter comes from [Johansson et al., 2019]. The key idea is to make use of the support sufficiency divergence to underline the effect of support overlap in domain adaptation.

Some more quantities are needed to write it, we will define them here. Firstly, a weighted loss on the source domain is needed, the weights have the following expression.

$$w^{\epsilon}_{\mathcal{S},\mathcal{T}}(x) = \begin{cases} \frac{\mathcal{T}(x)}{\mathcal{S}(x)} \text{ if } \mathcal{S}(x) \geq \epsilon \\ 1 \text{ otherwise} \end{cases}$$

On top of that, a term called *excess target information loss* is also involved in the bound. The hypothesis h is considered to be composed of two functions $h = f \circ g$ which will very much correspond to the IDR framework presented later.

$$\eta_g(f) = \mathbb{E}_{\mathcal{T}}\left[\Delta_{\mathcal{T},\mathcal{P}}(x) - \Delta_{\mathcal{S},\mathcal{P}}(x)\right]$$

where $\Delta_{\mathcal{D},\mathcal{P}}(x) = \mathbb{E}_{\mathcal{P}_{Y|g(X)}\ell(f(g(x)),y)} - \mathbb{E}_{\mathcal{D}_{Y|X}\ell(f(x),y)}$.
Once these quantities are introduced, we can state the following bound.

Theorem 3.5. *(Proof in [Johansson et al., 2019])*
Let h be a hypothesis composed of an encoder g and a head f. Then, the following holds.

$$R_T(h) \leq \mathbb{E}_{\mathcal{S}}[w^{\epsilon}_{\mathcal{S},\mathcal{T}}(x)\ell(f(g(x)), y)] + MD^{\epsilon}_{sup}(\mathcal{S}||\mathcal{T}) + \eta_g(f)$$

3.1.2 Unifying view on learning bounds

Naturally, someone interested in distribution shifts may wonder about the usefulness of these bounds in applied settings. Risk, distances, and other terms involved are ideal quantities that one can perhaps approximate but not compute perfectly. As a matter of fact, the quantities in the learning bounds are not all tractable. By tractable, we mean coherently approximated or bounded using learning pairs from the source domain and inputs from the target.

To start observing commonalities between the bounds, we see that in each of them, there exists an intractable term that would require access to complete learning pairs from the target domain. Secondly, we also directly see that each bound implies a statistical distance between the marginal distributions of the inputs $\mathcal{S}_X$ and $\mathcal{T}_X$. Obviously, the risk in the source domain appears directly or not within the bounds. If a model cannot treat the task in the source domain, it is very unlikely that it will work on target. From that, we deduce the three following criteria for unsupervised domain adaptation.

I **Performance on source** : A hypothesis with high risk on source cannot be expected to perform better on target. This aspect is tractable and depends directly on each hypothesis. Even if we cannot directly access the risk (as it is an expectation) we can consistently approximate it by empirical risk [Bartlett et al., 2021].

II **Covariate shift** : Inputs (or covariates) are the only quantities available on both domains. Therefore, one can quantify the shift between distributions through a statistical distance. Each statistical distance presented in the bounds can be approximated and we invite the reader to consult the associated references treating this.

III **Conditional shift** : Even if the two first metrics are low (the hypothesis performs well on the source and the covariates are distributed almost identically), we are not done yet. Indeed, to be sure our hypothesis will perform the same on target, it is obviously required that a pair of identical inputs from each domain receive the same label. In the deterministic labeling setting, it means that both labeling functions are the same. In the stochastic setting, we extend the criteria to conditional distributions $\mathcal{S}_{Y|X}$ and $\mathcal{T}_{Y|X}$. The crucial fact is that the criterion is always intractable as such because we have no knowledge even on $\mathcal{T}_Y$.

Bound	**I: Perf. on source**	**II: Cov. shift**	**III: Cond. shift**
3.1	$R_S(h)$	Total variation	Expected difference between labels
3.2	$R_S(h)$	$\mathcal{H}\Delta\mathcal{H}$-divergence	Minimal joint error
3.3	$R_S(h, h^*_S)$	Discrepancy	Mean disagreement between risk minimizers
3.4	$R_S(h)$	Wasserstein distance	Wasserstein distance between conditional distributions
3.5	$\mathbb{E}_{\mathcal{S}}[w^{\epsilon}_{\mathcal{S},\mathcal{T}}(x)\ell(h(x), y)]$	Domain sufficiency divergence	Excess target information loss

Table 2: Metrics involved in learning bounds.

3.2 Impossibility theorems

The learning bounds presented here are solely upper bounds on risk. They indicate quantities that ensure good performance if they are minimized. In other words, the bounds embody sufficient conditions for efficient learning. On the other hand, impossibility theorems show the necessity of several assumptions. These theorems form a set of constructively-proven results stating the necessity of several assumptions for DA to be possible. Here, the proofs hold for binary classification. We recall them here to introduce the theoretical basis of impossible DA, *i.e.* the existence of cases for which DA is impossible.

In [David et al., 2010], the authors identify three necessary assumptions that are similar to some extent to the criteria we introduced in the previous section. These are the following.

1. **Small joint optimal error** : Denoting the joint optimal error $\lambda_{\mathcal{H}} = \min_{h\in\mathcal{H}}[R_S(h)+R_T(h)]$, the authors require that this quantity be small (more precisions later). This represents both criteria I and III.

2. **Small discrepancy** : The distance between the covariate distributions should be small too. For the statement of the theorem, the authors use the $\mathcal{H}\Delta\mathcal{H}$-divergence. This is criterion II.

3. **Covariate shift** : Finally, the conditional distributions should be the same. Only the covariate distributions should differ. This is exactly the criterion III.

For the first theorem, it is shown that even if the joint optimal error is small, under the covariate shift assumption, we can construct a case where the classifier performs poorly on the target domain.

Theorem 3.6. *(First impossibility theorem, necessity of a small covariate shift) [David et al., 2010] Let $\mathcal{H}$ be a hypothesis class for binary classification. Let $\mathcal{S}$ be the source distribution. There exists a target distribution $\mathcal{T}$ and a hypothesis h verifying the following.*

1. $\lambda_{\mathcal{H}} \leq \epsilon$
2. $W(\mathcal{S}_{Y|X}, \mathcal{T}_{Y|X}) = 0$
3. $\mathbb{P}(\mathbb{E}_T[\ell^{01}(h(x), y)] \geq 1/2) \geq 1/2$, *with words, the probability that the error on target is greater than 1/2 is greater than 1/2.*

Afterwards, the second theorem shows the necessity of a small joint error. In practice, this means that there exists a hypothesis that works well on the source and target domains.

Theorem 3.7. *(Second impossibility theorem, necessity of a small joint error) [David et al., 2010] Let $\mathcal{H}$ be a hypothesis class for binary classification. Let $\mathcal{S}$ be the source distribution. There exists a target distribution $\mathcal{T}$ and a hypothesis h verifying the following.*

1. $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S}_X, \mathcal{T}_X) \leq \epsilon$
2. $W(\mathcal{S}_{Y|X}, \mathcal{T}_{Y|X}) = 0$
3. $\mathbb{P}(\mathbb{E}_T[\ell^{01}(h(x), y)] \geq 1/2) \geq 1/2$, *with words, the probability that the error on target is greater than 1/2 is greater than 1/2.*

It emphasizes that one cannot be sure of the performance of its learning method even under the covariate assumption if the joint error or covariate shift is not controlled. These two theorems are shown for binary classification, but we will consider that they extend to any task.

3.3 Covariate shift assumption

Since the beginning of this work, we have discussed the hypothesis of covariate shift. It is a restriction on the possible shift between the distributions. Namely, it means the equality between conditional distributions, $\mathcal{S}_{Y|X} = \mathcal{T}_{Y|X}$. This means that only the distributions of the covariate should differ, hence the name.

Intuitively, these assumptions are not extremely restrictive. It indicates that if a target input is exactly the same as the source one, it should receive the same output. If we suppose that there exists an exact labeling function, we are under covariate shift.

We can better understand the assumption by saying that if a target input $x_T \sim \mathcal{T}_X$ could have been sampled from the source distribution (due to overlapping supports of $\mathcal{S}_X$ and $\mathcal{T}_X$), the distribution of its output has to be identically distributed as if it had come from the source distribution $\mathcal{S}_X$.

For instance, the example with Louvain-La-Neuve and Boston housings from the introduction is probably not verifying the covariate shift assumption. Indeed, if one considers student efficiency in Louvain-La-Neuve and finds the exact equivalent in Boston, they will probably not share the same price. For now, strict equality between the two marginal distributions is considered. It is possible to relax it to equality in terms of effective support.

As the example of housing does not verify the covariate shift, let us consider another one. A simple example related to computer vision that has been treated in some surveys [Csurka, 2017] is a source domain made of pictures and a target domain made of drawings. An example of this is shown in figure 9 with animal classification[1].

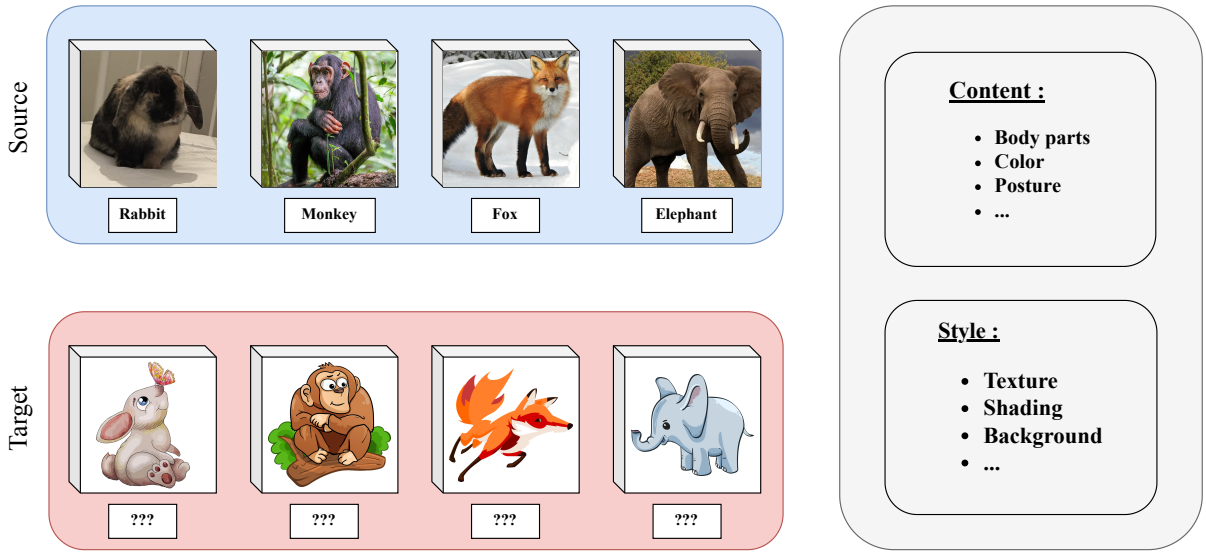


Figure 9: A well-known example of covariate shift in computer vision.

[1]All images are royalty-free and coming from https://pixabay.com

Through the learning bounds and the impossibility theorems, we saw that the covariate shift is a helpful assumption that increases our hope for a successful DA. However, it is not sufficient. One cannot imagine solving any domain adaptation problem by only assuming covariate shift. A source domain made of animal pictures will not be able to classify hand-written digits. In this case, the distribution of the labels is not the same. But even requiring this is not enough. Let us imagine a source domain of hand-written Arabic numerals and a target domain of hand-written Latin numerals. The labels share the same distribution, but it is impossible for the model to adapt the domains. We will identify this as the content shift.

3.3.1 Generative process with covariate shift

We will now spend some time modeling a generative process that will help us understand examples of covariate shift. This generative process will be designed in such a way that the conditions of the covariate shift are verified, which provides intuitive interpretation of the methods that will be described afterwards. On top of this, this model will also help us define the limitations of domain adaptation.

We consider that a sample x is generated by two latent variables called style and content. The content, denoted κ is defined by its informativity regarding the outcome y and, therefore, its domain-invariance. On the other hand, the style, denoted ς is defined by its informativity regarding the domain and its independence with respect to y. The input x is therefore obtained as a function of κ and ς. The generative process is illustrated by a Markov chain represented on figure 10. Similar ideas about generative processes can be found in the literature of UDA [Kong et al., 2023, Yue et al., 2023].

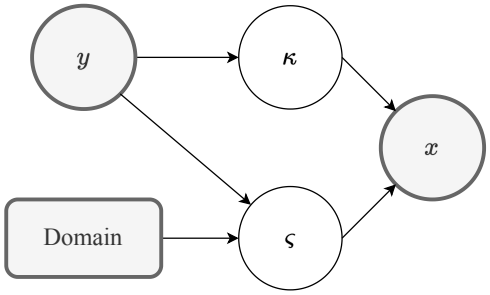


Figure 10: Generative process of the data under the assumptions of covariate shift.

To maybe provide more intuition on what the interpretation of style and content is, one can see on figure 9 that the style is composed of aspects related, especially to the domain, while the content contains information about the class. From that, we can deduce that only the style component is affected by the domain and is irrelevant regarding the task. Considering how methods insert inside this will be treated later, we can already tell that disentangling the style and content components of the data may become a key aspect of the UDA methods.

3.3.2 Covariate shift and inverse problems

When considering IP solving with DA, we can also establish a link with the covariate shift assumption. Here, our covariates are the observations. These are influenced by two objects

: the observation operator and the hidden state. As our goal is to recover this hidden state, this can be identified as the content. The latter is altered by the observation operator, and we would require our model to be able to bypass this alteration. Consequently, the operator represents the style.

Additionally, we can extend this with the notion of dictionary learning, the content component corresponds thus to the representations of the instance in the dictionary of interest.

This means that we can encounter DA under covariate shift with inverse problems by shifting the operator.

$$x_S = A_S y_S$$
$$x_T = A_T y_T$$

Recalling the two fundamental aspects of IP : data consistency and prior regularization, one can use UDA as an attempt to extract the prior (in the dictionary) from the source domain to regularize the target.

3.3.3 Learning bounds under covariate shift

About learning bounds, we observed that they are three-fold. Two aspects are directly observed from the dataset and the choice of model. The last term, related to conditional shift, is always intractable. Introducing the covariate shift assumption causes this term to vanish. This means that the learning bounds consist now only of a term related to the performance of the source and a term related to the distribution shift of the covariates.

4 Methods based on invariant and discriminative representations

The goal of this work is to provide the most overall view of UDA. Regarding methods, we require them to be as versatile as possible (valid for regression and classification tasks and for any data modality). Also, we want the panel of chosen methods to be as representative as possible of the state of the art. Through the literature, many classifications of UDA methods have been proposed depending on their seminal ideas. We make the original choice to unify all of them under the same flexible paradigm. We name the latter a group of methods based on invariant and discriminative representations (IDR).

Below, we describe seven methods that have been introduced over the last decade. Even if not all of them have been presented as such in the original papers, we present here a unifying view through the IDR framework, with slight adaptation in some cases.

- **Expressivity and capacity of deep learning** : The superiority of neural networks for machine learning tasks is not to be shown anymore. Thanks to their expressivity and capacity to learn complex patterns, they are currently the most appealing tools for generic machine learning tasks.

- **Versatility** : Often, UDA has been considered for classification tasks. Using it for regression has hardly ever been put forward. Notwithstanding, the IDR framework can also be applied to regression tasks.

- **Easily comparable** : Our objective in this work is not to compete with the state-of-the-art results of unsupervised domain adaptation. We rather aim at comparing several seminal approaches to provide a better understanding of the underlying principles of UDA. For this reason, we do not need to overcomplicate the neural network blocks or leverage foundations networks. That being said, we can easily compare the methods by conserving the same architecture for head and encoder networks and using similar optimization procedures.
- **Common failure** : We are sorry to maybe disappoint the hopeful reader, but we will see further that the methods presented here are unable to tackle the distribution shift in some specific cases. Following our interpretation, this is due to the common structure of IDR methods. Being provided with this unifying view, it is natural to deduce that all methods may fail without showing it for every specific case.
- **Interpretability** : Despite their expressivity, the major drawback of deep learning tools is their lack of interpretability. It is almost impossible to predict or understand how the intermediate representations are built. If one ends up with unexpected results, it is difficult to understand why. However, with the IDR framework, we gain a bit more interpretability. One should expect the distribution to be aligned after the encoder, and this can be checked by computing various divergences or by visualizing the latent space (with possibly dimensionality reduction techniques (PCA, t-SNE, *etc.*)).

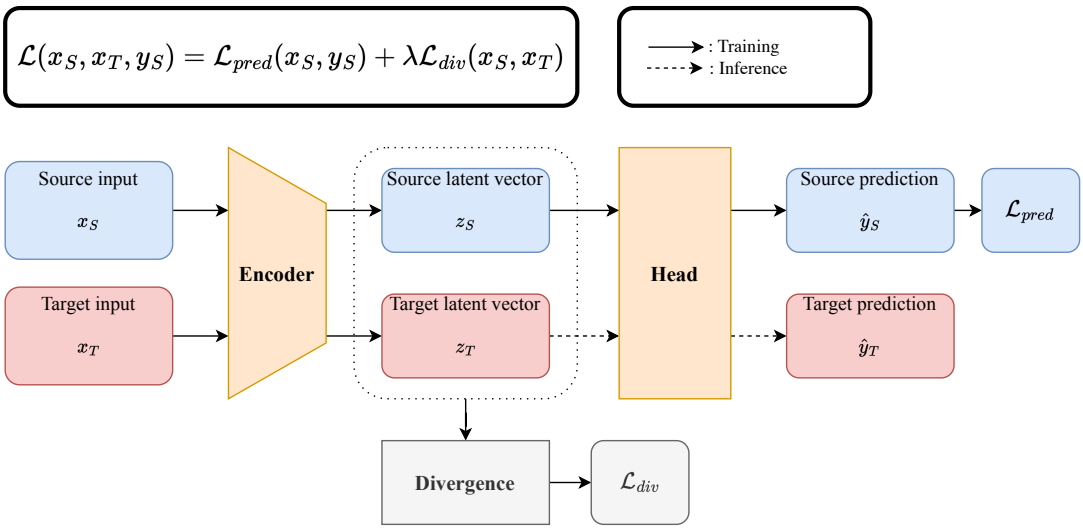


Figure 11: Illustration of the methods based on invariant and discriminative representations.

The architecture of this type of method is mainly composed of two parts. An encoder aims at recovering a representation of the content, and then a classifier takes the output of the encoder, called a latent vector and denoted z, as input. The global structure is represented in figure 11. The vectors z are the representations that are expected to be domain-invariant and discriminative. They should be identically distributed across domains and contain all the information needed to perform the task.

At training time, source and target data x_S, x_T are fed to the encoder to return z_S, z_T. Then, with a given method, a given divergence between the empirical distributions of the latent vectors is computed. Afterwards, the latent vector coming from the source domain is fed to the head to approximate the output y_S. The total loss is then a weighted sum of the divergence loss $\mathcal{L}_{div}$ and the prediction loss $\mathcal{L}_{pred}$. The divergence loss is expected to be inversely proportional to the divergence.

Later, when we will be observing the results of the methods, it will be fundamental to compare them to the naive option, which is training the model on source only and then testing it on target. In this context, the unlabeled data from the target is completely useless. The interest of UDA is solely to leverage this data to improve the model. That baseline model is represented in figure 12 as a comparison of the diagram that we will introduce later in the section.

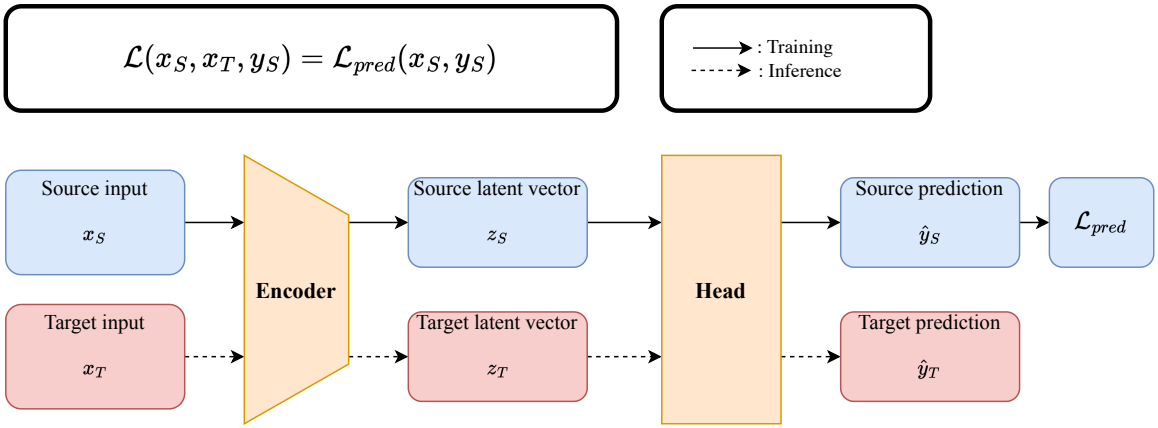


Figure 12: Naive baseline for UDA.

In table 3, we provide a list of the considered methods of this work. In the literature, IDR methods probably correspond to what is usually called divergence or discrepancy-based domain adaptation. Yet, some methods like DANN or DeepJDOT are never presented as such, but we unify them following the IDR framework.

Method	Divergence	Training approach	References
Deep Correlation Alignment (DeepCORAL)	Frobenius distance between correlation matrices	Direct minimization	[Sun and Saenko, 2016]
Deep Domain Confusion (DDC)	Maximum Mean Discrepancy	Direct minimization	[Tzeng et al., 2014]
Deep Adaptation Network (DAN)	Multi-Kernel Maximum Mean Discrepancy	Direct minimization	[Long et al., 2015]
Kullback-Leibler Guided Domain Adaptation (KLGDA)	Kullback-Leibler divergence	Variational inference	[Nguyen et al., 2021a]
Domain Adversarial Neural Network (DANN)	Jensen-Shannon divergence	Adversarial training	[Ganin et al., 2016]
Wasserstein Distance Guided Representation Learning (WDGRL)	Wasserstein distance	Adversarial training	[Shen et al., 2018]
Deep Joint Distribution Optimal Transport (DeepJDOT)	Wasserstein distance	Optimal Transport Wasserstein minimization	[Damodaran et al., 2018]

Table 3: Methods based on invariant and discriminative representations.

To be consistent and to inform the curious reader, a list of methods that have not been considered in this work and why they have not been taken into account is provided in appendix A.

4.1 Descriptions of methods

The principal difference between the methods in this category can be seen as the choice of divergence and how it is computed. Now, we will describe the methods in more detail.

4.1.1 Deep Correlation Alignment (DeepCORAL)

Let us start softly with an easy-to-understand method. Two distributions are the same if and only if all of their moments match. For this reason, early methods of domain adaptation developed the idea of constructing a transformation such that source and target

distributions have identical moments. This was the idea behind CORAL [Sun et al., 2016] which takes into account the second-order statistics of the data. The latter has been extended to the deep learning framework with DeepCORAL [Sun and Saenko, 2016]. It takes the typical form of an IDR method, where the divergence is the Frobenius distance between the correlation matrices of the latent vectors. Contrary to the original CORAL, we do not directly construct a transformation that makes the moment match while ignoring the influence of the task. Here, this transformation is replaced by the encoder, which still needs to extract discriminative features. The covariance matrices on source and target domains can be easily computed by the empirical estimates.

$$\mathcal{L}_{div} = ||\Sigma_S - \Sigma_T||_F^2$$

If we denote by $\mathfrak{S}$ and $\mathfrak{T}$ the datasets of source and target domains (in practice, the considered batch), and Z_S and Z_T the stacked latent vectors, the covariance matrices are computed as follows.

$$\Sigma_S = \frac{1}{|\mathfrak{S}|-1}(Z_S^\top Z_S - \frac{1}{|\mathfrak{S}|}Z_S^\top \mathbf{1}\mathbf{1}^\top Z_S), \quad \Sigma_T = \frac{1}{|\mathfrak{T}|-1}(Z_T^\top Z_T - \frac{1}{|\mathfrak{T}|}Z_T^\top \mathbf{1}\mathbf{1}^\top Z_T)$$

We will later observe that this method lacks efficiency. One way to interpret this is the weakness of the divergence used. Indeed, two distributions can have the same covariance matrix but be very different with respect to the classical metrics (Kullback-Leibler, Wasserstein, *etc.*).

4.1.2 Deep Domain Confusion (DDC)

This method has been introduced in [Tzeng et al., 2014] as one of the first deep UDA methods (where DeepCORAL is an extension of a shallow one). Here, the domain alignment is done by employing the MMD introduced in section 2.3.2. We recall that the seminal idea of this metric is testing the hypothesis that two samples come from the same distribution. Therefore, using MMD is probably a good idea to align domains in the latent space of the encoder. Interestingly enough, the MMD is a divergence that can be directly computed without requiring any further steps (training a neural network, solving a constrained optimization problem, *etc.*). It can be computed following the biased or unbiased estimates from definitions 2.7 and 2.8.

Regarding the kernel choice, we considered two types of kernels. Firstly, the linear one $k_L(\cdot,\cdot)$ which simply corresponds to the dot product between vectors. Secondly, the Gaussian one $k_G(\cdot,\cdot)$ with different bandwidth γ_i. For this latter choice, we considered a sum of several kernels to increase the expressivity of the metric.

$$\begin{aligned} k_L(x,x') &= x^T x' \\ k_G(x,x') &= \sum_{i=0}^{n_K-1} \exp\left(-\frac{||x-x'||^2}{2\gamma_i}\right) \end{aligned}$$

Regarding the choice of estimate (biased or not), both have been tested, and no significant difference has been observed. Therefore, we kept its unbiased version.

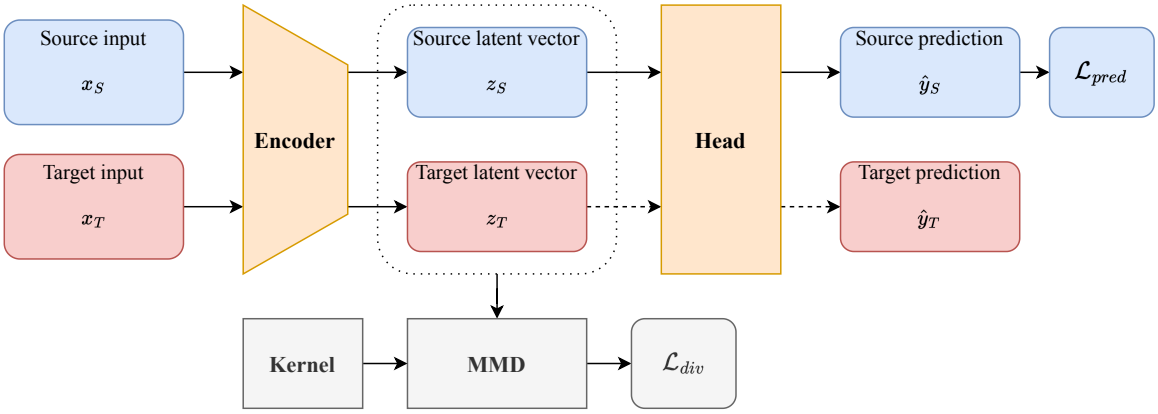


Figure 13: Modification of the IDR method structure for DDC.

4.1.3 Deep Adaptation Network (DAN)

Following almost the same procedure as for DDC, we now consider the multi-kernel version of the MMD [Long et al., 2015]. But this time, our kernel will change during the training phase. More precisely, it is a weighted sum of differently parameterized kernels from the same family (here Gaussian kernels with different bandwidth). The weights are updated during the training process for the MMD to be more sensitive to the distribution shift. As a matter of fact, DAN training can be seen as a minmax optimization problem. Indeed, recalling the optimization problem yielding the kernel update for MK-MMD (equation 6), we have the following objective.

$$\min_{\theta_{\text{enc}} \in \Theta_{\text{enc}}} \max_{k \in \mathcal{K}} \frac{MMD^2(\mathcal{S}_Z, \mathcal{T}_Z)}{\sigma_k^2}$$

where $\theta_{\text{enc}} \in \Theta_{\text{enc}}$ corresponds to the parameters of the encoder. In a practical setting, the kernel update is done by convex optimization of the problem exposed in equation 7.

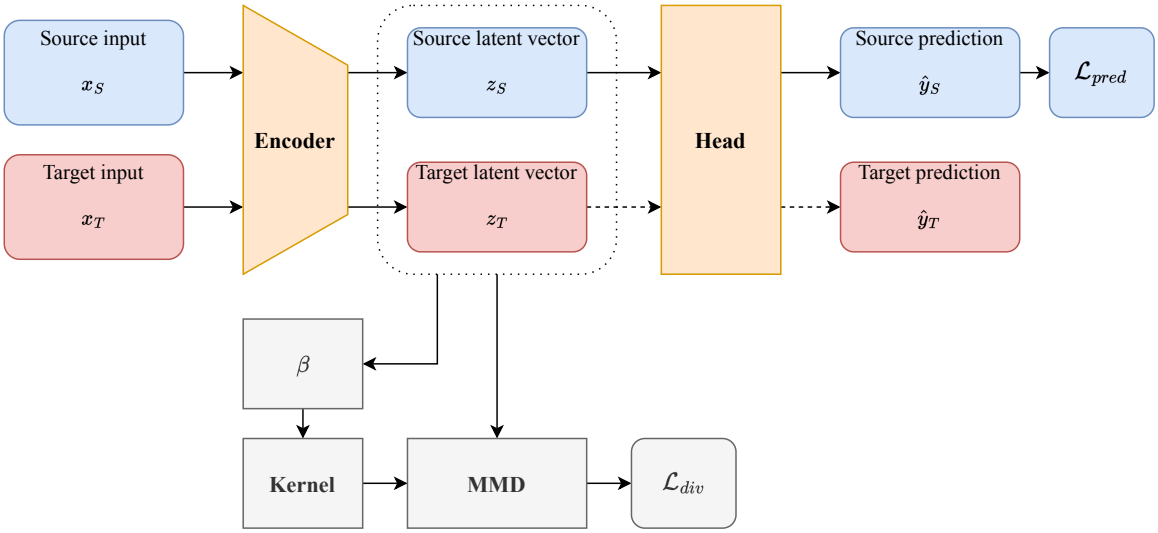


Figure 14: Modification of the IDR method structure for DAN.

4.1.4 Kullback-Leibler Guided Domain Adaptation (KL-GDA)

When thinking about a metric quantifying how two distributions differ, the information-theoretic divergences (Kullback-Leibler and Jensen-Shannon) may be among the first ones that come to mind. Unfortunately, we saw earlier that none of them are suitable when considering empirical distributions. For this reason, we need to enter the variational framework. In this setting, we will not consider the empirical distributions $\hat{\mathcal{S}}_Z = \sum_{z \in \mathfrak{S}_Z} \delta_{z_i^S}$ and $\hat{\mathcal{T}}_Z = \sum_{z \in \mathfrak{T}_Z} \delta_{z_i^T}$ with zero density almost everywhere. We will rely on other approximations detailed below. Once, this is done, one can approximate the KL-divergence, this method has been introduced in [Nguyen et al., 2021a].

Through the elaboration of a new learning bound involving KL divergence, the authors showed the necessity to reduce the divergence $\mathrm{KL}(\mathcal{T}_Z||\mathcal{S}_Z)$ We refer the reader to the original paper for details on this bound. The idea of using also reverse divergence $\mathrm{KL}(\mathcal{S}_Z||\mathcal{T}_Z)$ is discussed as it disposes of a zero-forcing effect. Indeed, it will force the target distribution to be null out of the support of the source distribution. This can be relevant in cases of partial domain adaptation. As we do not pay particular attention to this specific case, we will not detail it further and not consider the use of reverse divergence for KL-GDA.

For this variational approach, one needs to arbitrarily choose a parametric posterior distribution $\mathcal{Q}(z|x)$. In other words, instead of computing the code z corresponding to the input x, the encoder will compute the parameters of the posterior distribution. The common choice is to consider a normal distribution with a diagonal covariance matrix.

$$\mathcal{Q}(z|x) = \mathcal{N}(\mu(x), \Sigma(x))$$

Afterwards, we still need to approximate the marginal distributions $\mathcal{S}_Z(z) = \mathbb{E}_S \mathcal{S}_{Z|X}(z|x)$ and $\mathcal{T}_Z = \mathbb{E}_T \mathcal{T}_{Z|X}(z|x)$. Due to our choice of posterior distribution, the marginals are gaussian mixtures with $\mathcal{S}_X$ and $\mathcal{T}_X$ being the mixture distributions.

Computing the Kullback-Leibler divergence between these two mixtures can be done in two ways. The first one is more intuitive and relies on considering empirical distributions for $\mathcal{S}_X$ and $\mathcal{T}_X$. Conversely, the second one may be considered more sophisticated and relies on upper and lower bounds that have been derived for Gaussian mixtures. They were derived in [Durrieu et al., 2012]. Let's first consider the empirical mixture approach. The marginals are approximated by the following expressions :

$$\hat{\mathcal{S}}_Z(z) = \frac{1}{|\mathfrak{S}|} \sum_{x \in \mathfrak{S}} \mathcal{Q}(z|x)$$
$$\hat{\mathcal{T}}_Z(z) = \frac{1}{|\mathfrak{T}|} \sum_{x \in \mathfrak{T}} \mathcal{Q}(z|x)$$

Finally, the KL divergence is computed by averaging the logarithms of the densities.

$$\mathcal{L}_{div} = \hat{\mathrm{KL}}(\hat{\mathcal{T}}_Z, \hat{\mathcal{S}}_Z) = \frac{1}{|\mathfrak{T}|} \sum_{z \in \mathfrak{T}_Z} \log \hat{\mathcal{T}}_Z(z) - \log \hat{\mathcal{S}}_Z(z)$$

Concerning the second approximation, we will not detail it thoroughly here. The global idea is to simplify the computation by leveraging Jensen's inequality and to make apparent the pairwise divergences between the Gaussian composing each mixture. In this way, an upper and lower bound are derived, and the divergence is approximated by their means.

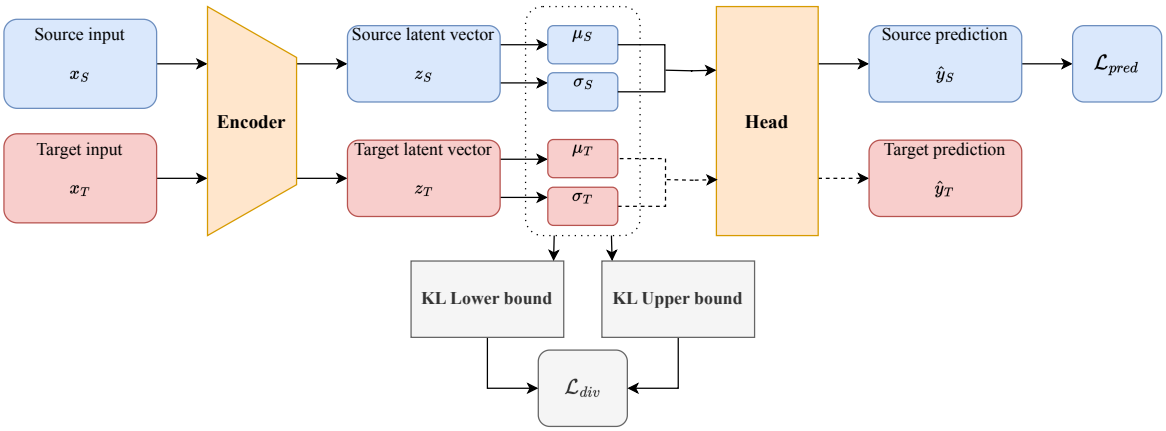


Figure 15: Modification of the IDR method structure for KL-GDA.

4.1.5 Domain Adversarial Neural Network (DANN)

Given two distributions, an interesting scheme to measure how different they are is to observe if a neural network can learn to separate them. Being provided with a training set, will the NN be able to tell if an unseen data point comes from the source or the target distribution? If the NN is unable to perform this binary classification, it probably means that the domains are aligned. This is the idea behind the domain adversarial neural network (DANN) [Ganin and Lempitsky, 2015]. The additional network dedicated to this task is called the domain discriminator, denoted DISC. This function is trained to approximate $\mathsf{DISC}^\dagger$.

$$\mathsf{DISC}^\dagger = \operatorname{argmin}_{\mathsf{DISC}} \mathbb{E}_{z\sim\mathcal{S}} \log \mathsf{DISC}(z) + \mathbb{E}_{z\sim\mathcal{T}} \log(1 - \mathsf{DISC}(z))$$

$$\mathsf{DISC} = \operatorname{argmin}_{\mathsf{DISC}\in\mathcal{F}_{\text{disc}}} \underbrace{\sum_{z\in\mathfrak{S}} \log \mathsf{DISC}(z) + \sum_{z\in\mathfrak{T}} \log(1 - \mathsf{DISC}(z))}_{\mathcal{L}_{\text{disc}}(\mathfrak{S},\mathfrak{T})}$$

One interesting benefit of this metric is its scalability. The discriminator capacity can be tuned arbitrarily to provide coarser or finer information on the domain alignment. In the same way, the discriminator can be adapted to the modality of the latent space. For instance, if the encoder is only composed of convolutional layers, it is probably a good idea to use a convolutional discriminator.

The divergence term in the loss is represented by the opposite of the loss of the discriminator, $\mathcal{L}_{\text{div}} = -\mathcal{L}_{\text{disc}}$. The whole method is considered adversarial because different parts of the network are in competition, maximizing and minimizing the same quantities. While the discriminator tries to distinguish domains, the encoder tries to fool it by making the latent vectors indistinguishable.

Usually, adversarial networks are trained sequentially in a two-step process. One gradient is backpropagated through the related part of the network, while the other part is frozen. Then, the opposite gradient is backpropagated through the other part of the network. This process is repeated until convergence. Unfortunately, this is known to be possibly unstable and difficult to hyperparameterize. In [Ganin et al., 2016], the authors proposed a way to train the network in a single step by using the gradient reversal layer (GRL).

$$\mathrm{GRL}(x) = \begin{cases} x & \text{forward-propagation} \\ -x & \text{backward-propagation} \end{cases}$$

Recalling what has been discussed about Jensen-Shannon divergence in subsection 2.3.1, one can relate the objective of the discriminator as an approximation of the Jensen-Shannon divergence.

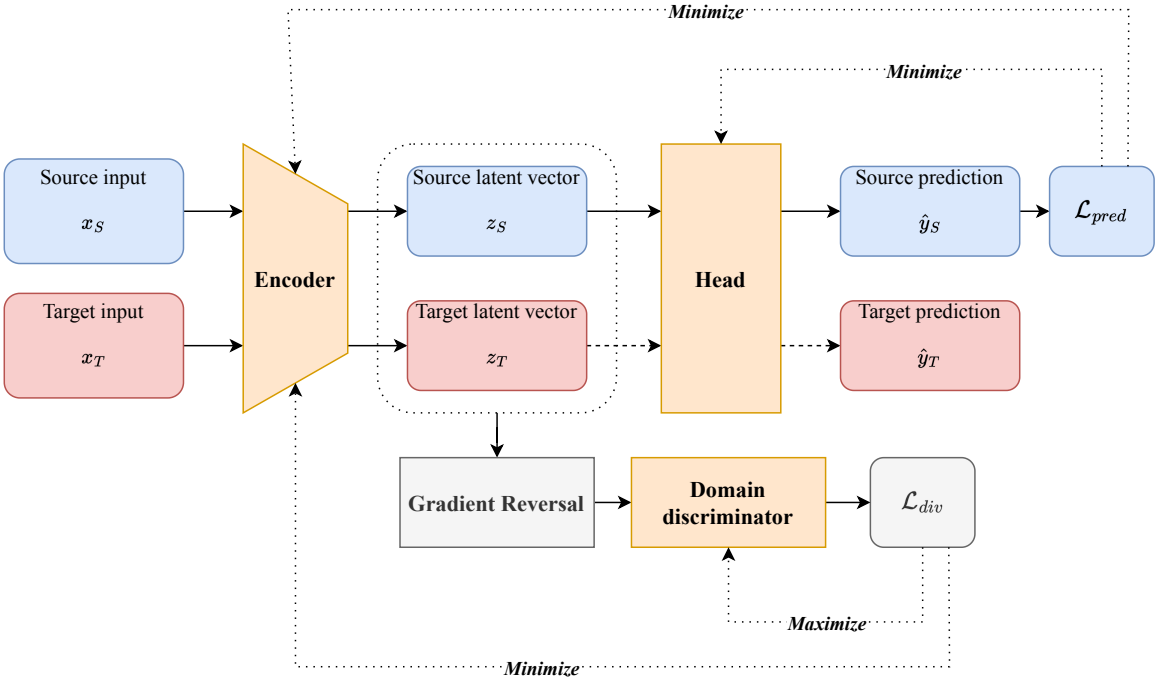


Figure 16: Modification of the IDR method structure for DANN. The GRL induces the adversarial aspect of training. Dotted arrows represent what objectives are maximized or minimized by the different parts of the network.

Domain Adversarial U-Network Later in this work, we will work on a computational imaging example. In this context, we adapted the structure of the U-Net to the DANN framework. This takes the form represented in figure 17. It consists simply of a specific choice of encoder and head but remains totally in the IDR concept. Due to the skip connections of the original architecture, one must put the discriminator at the shallowest level of the encoder to enforce invariance at all levels. In this case, we will prefer to use a convolutional discriminator.

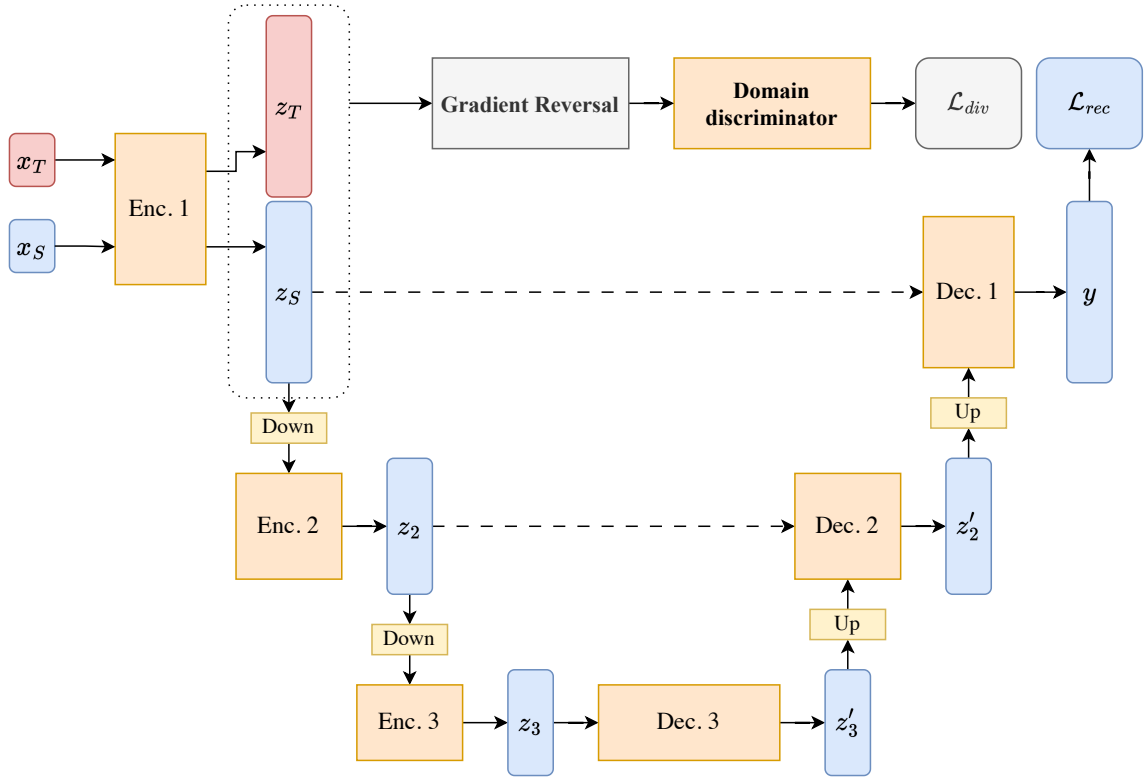


Figure 17: Adaptation of the U-Net architecture to the IDR framework.

4.1.6 Wasserstein Distance Guided Representation Learning (WDGRL)

Minimizing Wasserstein distance has shown its relevance through its presence in learning bounds. However, one can deduce from its formal definition that it cannot always be explicitly and easily computed. It requires solving an optimization problem where constraints imply distributions from which we only have access to samples. Equivalently, it can be computed through the Kantorovich-Rubinstein duality, which also involves optimization procedures. Inserting this distance as a divergence measure in our IDR methods may not be straightforward. Indeed, it would cause two optimization problems (the learning procedure and the Wasserstein distance computation) to be solved simultaneously. This will be achieved by two UDA methods that we are now presenting.

Firstly, the authors of [Arjovsky et al., 2017] proposed to use a neural network, called the critic CRIT, to compute the distance. This technique does not follow from the primal definition but from the KR dual definition (theorem 2.4). In the latter, we search a Lipschitz-ball of a Banach space for a function that magnifies the difference between the two distributions. The neural approximated Wasserstein restricts the Banach space to a neural network architecture $\mathcal{F}_{\text{critic}}$. The functional form maximized by the critic is the one appearing in the Kantorovich-Rubinstein duality. This yields the following expression for the distance.

$$W_{\mathcal{F}}(\mathfrak{S}_X, \mathfrak{T}_X) = \sup_{\mathsf{CRIT} \in \bar{\mathcal{F}}_{\text{critic}}} \frac{1}{|\mathfrak{S}||\mathfrak{T}|} \left| \sum_{x \in \mathfrak{S}} \mathsf{CRIT}(x) - \sum_{x \in \mathfrak{T}} \mathsf{CRIT}(x) \right|$$

The set $\bar{\mathcal{F}}_{\text{critic}} \subset \mathcal{F}_{\text{critic}}$ stands for the Lipschitz constraint. One has several possibilities for imposing the latter. However, the first way proposed was weight clipping, gradient penalty [Gulrajani et al., 2017] is much more common now. Clipping produces vanishing or exploding gradients.

$$W_{\mathcal{F}}(\mathfrak{S}_X, \mathfrak{T}_X) = \sup_{\mathsf{CRIT} \in \mathcal{F}_{\text{critic}}} \frac{1}{|\mathfrak{S}||\mathfrak{T}|} \left| \sum_{x \in \mathfrak{S}_X} \mathsf{CRIT}(x) - \sum_{x \in \mathfrak{T}_X} \mathsf{CRIT}(x) \right| + \lambda \sum_{x \in \mathfrak{D}^+} ||\nabla \mathsf{CRIT}(x) - 1||_2^2$$

The set $\mathfrak{D}^+$ is an augmented dataset containing $\mathfrak{S}$, $\mathfrak{T}$ and convex combinations of them. Even if this penalty does not enforce inequality but equality, it has shown its efficiency.

Regarding practical aspects, this approach is also adversarial due to the interaction between the domain alignment term, which is minimized, and the maximization appearing in the Kantorovich-Rubinstein duality. The critic is trained to magnify the differences between the distributions, while the encoder is trained to align them.

Because of the penalty term for the critic, one cannot employ the gradient reversal layer as in DANN to stabilize the training procedure.

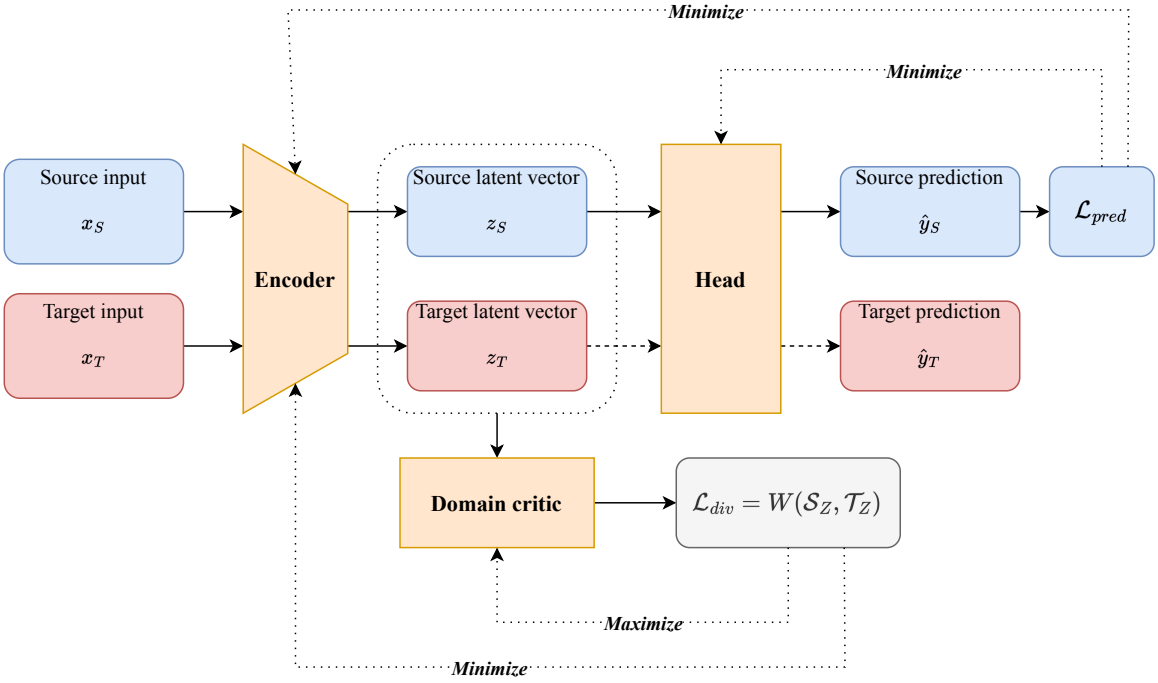


Figure 18: Modification of the IDR method structure for WDGRL. Dotted arrows represent what objectives is maximized or minimized by the different parts of the network.

Interestingly enough, the couple of adversarial methods for UDA, DANN and WDGRL is analogous to the couple of adversarial methods for generative deep learning, Generative Adversarial Network (GAN) [Goodfellow et al., 2014] and Wasserstein-GAN (WGAN) [Arjovsky et al., 2017]. Instead of aligning two sampled distributions, generative models learn to reproduce a sampled distribution with an artificial one. In this case, it is also needed to quantify the distance between two distributions. GANs leverage the

loss of a discriminator like DANN and WGAN uses the neural approximated Wasserstein like WDGRL.

4.1.7 Deep Joint Distribution Optimal Transportation (DeepJDOT)

As introduced when discussing WDGRL, the Wasserstein distance is an appealing metric for domain adaptation but needs to be cleverly approximated. Just above, the distance was approximated from its dual form by restricting the Lipschitz unit ball to a set of neural networks with a gradient penalty. The method that we present now will rely on its primal form and, therefore, involves constrained optimization.

Let's first gradually introduce the optimal-transport based DA methods to better understand the structure of DeepJDOT. It will be surprising to notice that, starting with different intentions from previous IDR methods, OT-based methods finally converge to comparable structures.

The seminal idea in leveraging OT (firstly explored in [Courty et al., 2017b]) for UDA is using the transport map to introduce pseudo-labels in the target domain. From the inputs of source and target domains, the OT-solver will output a transport map γ. The i-th input of the target domain $x_{T,i}$ is transported to the j-th source input $x_{S,j}$ with a probability $w_{ij} = \frac{\gamma_{ij}}{\sum_j \gamma_{ij}}$. Accordingly, the pseudo-label of $x_{T,i}$ is the weighted sum of the labels of $x_{S,j}$.

$$\tilde{y}_{T,i} = \sum_j w_{ij} y_{S,j}$$

The fundamental implicit assumption for this method to work is that the metric involved in sample spaces is such that the transport map conserves semantic information. In other words, one should expect the mapping to link samples sharing the same label. An example of such a convenient optimal transport is represented in figure 19. To end up in this favorable situation, one needs the metric on the sample spaces $d_X(x_S, x_T)$ to be meaningful regarding the task. Up to now, we have no specific reason to think that it is the case. Typically, for high-dimensional instances like images of figure 19, the usual distances (euclidean, cityblocks, *etc.*) will not be especially small between images from the same class[2]. One way to formulate this is by asking that the chosen distance between inputs is somehow proportional to the distance between outputs.

$$d_X(x_S, x_T) \propto d_Y(y_S, y_T)$$

That being done, our target dataset receives its pseudo-labels, and then we can train our model as if it were a regular supervised learning problem.

To circumvent this issue, a first idea proposed in [Courty et al., 2017a] is to leverage the joint distribution of the inputs and outputs to solve the optimal transport. This means that the metric $d_X(x_S, x_T)$ is replaced by something of the form $d_{XY}(x_S, x_T, y_S, y_T) = d_X(x_S, x_T) + \alpha d_Y(y_S, y_T)$. The careful reader should now notice that this should be illegal

[2]In the original paper, this was tackled by leveraging foundation networks to encode images as low-dimensional vectors of semantic features. For the sake of independence from these networks, we do not consider this alternative.

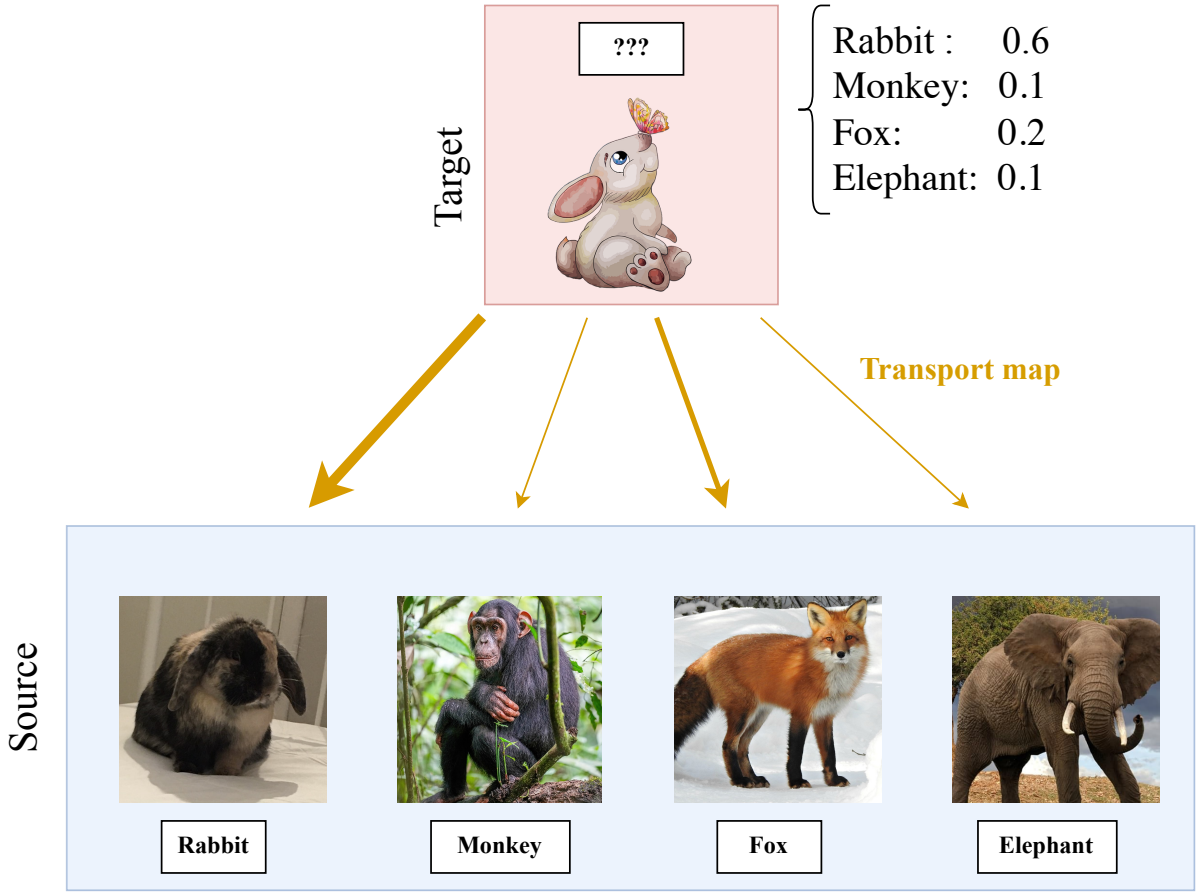


Figure 19: A convenient transport map for the covariate shift example of figure 9.

in the UDA framework as y_T is never accessible. In the Joint Distribution Optimal Transport (JDOT) method, the authors proposed to replace the labels with the predictions of the model and to jointly optimize the transport map and the model. The joint distributions of the inputs and the pseudo-labels generated by a model f are denoted $\mathcal{T}^f_{X,Y}$. The formulation becomes the following.

$$\min_{f\in\mathcal{F},\gamma\in\Gamma} d_{XY}(x_{S,i},x_{T,j},y_{S,i},f(x_{T,j}))\gamma_{ij} = \min_{f\in\mathcal{F}} W_1(\mathcal{S}_{X,Y},\mathcal{T}^f_{X,Y})$$

Please note, that this expression may seem similar to what we've done before, as the statistical distance is minimized. Yet, we usually tune the encoder for this minimization, here, that is the head component of the model that minimizes the objective. We will not go into details as JDOT is not the final method we consider, but the minimization over the hypothesis space $\mathcal{F}$ is free and can be done by any method (linear regression or neural network, for example).

Considering joint distribution may have solved part of the issue of finding a semantic metric, but we can still improve it. We would like the distance d_X to represent how close inputs are from a high-level perspective and not pixel-wise anymore. This is the idea behind the DeepJDOT method [Damodaran et al., 2018]. The distance between inputs is computed in the latent space of an encoder network. Supposing that this encoder extracts

meaningful features (for image, *e.g.* presence of given shapes or color patterns), simple distances like euclidean or cosine for d_Z become as relevant as in the low-dimensional case.

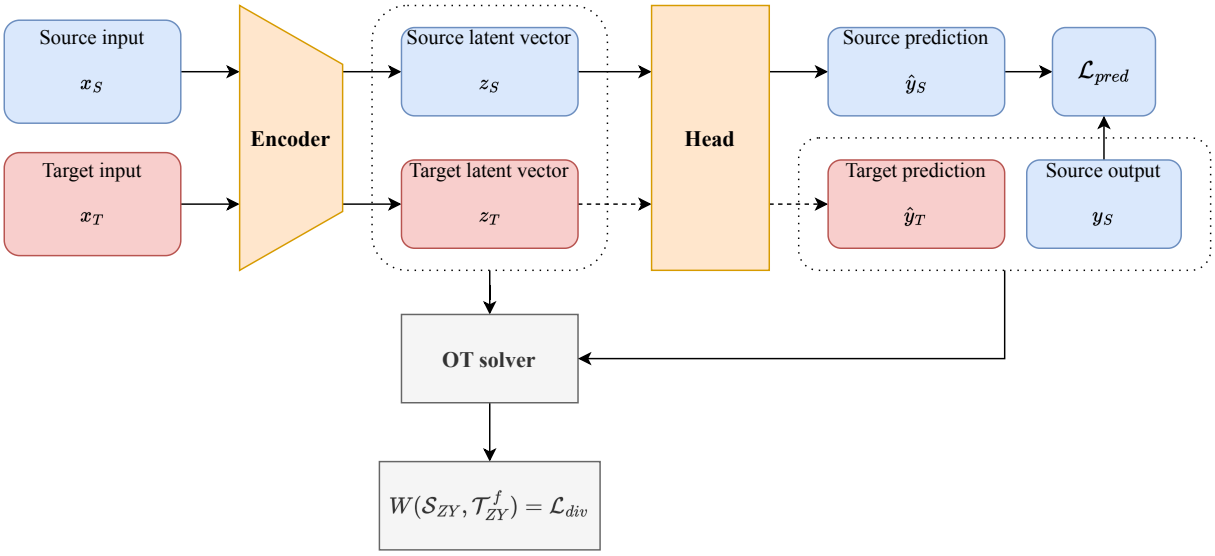


Figure 20: Modification of the IDR method structure for DeepJDOT.

More precisely, the inputs from the source and target domains x_S, x_T are fed to the encoder to return the latent representation z_S, z_T. For z_S, we dispose of the related label y_S and, for z_T, we compute the pseudo-label by passing it through the head. We obtain the transport map by solving the OT problem.

$$W(\mathcal{S}_{ZY}, \mathcal{T}^f_{ZY}) = \min_{\gamma \in \Gamma} \sum_{i,j} \big(d_Z(z_{S,i}, z_{T,j}) + \alpha d_Y(y_{S,i}, \mathsf{HEAD}(z_{T,j}))\big)\gamma_{ij}$$

Once the transport map is computed, the obtained Wasserstein distance and the prediction loss from the source domain are combined and backward-propagated to the encoder and the head. While WDGRL was an adversarial Wasserstein-guided method, DeepJDOT can be seen as a coordinate descent Wasserstein-guided method. Indeed, we sequentially update the neural network weights and the optimal transport map, which both minimize the same objective. This must be done sequentially, as the mapping γ must fulfill the constraints of the OT problem (represented by the set Γ).

Finally, one could be worried by the computational cost of solving an OT problem many times during the training. In a practical setting, we do not use the whole distribution of the dataset but only the minibatches. In other words, we compute transport maps only between randomly selected subsamples of the data. The smaller we take the batches, the farther we are from the optimal solution, but the faster the computation is. Therefore, this is a trade-off that must be adjusted.

4.2 Theoretical analysis

Up until now, we introduced the methods through intuitive thinking, and we relied on the black-box of deep learning to meet our expectations. Although we can leverage the

statistical learning tools introduced before to better understand this family of methods, more specifically, we will try to understand how the methods based on invariant and discriminative representations can lower the learning bounds.

Neural networks transform the input distribution into a sequence of intermediate hidden distributions. For that reason, learning bounds are not only valid for considering pairs of input x and output y but also for pairs of intermediate representations z and output. In this case, the hypothesis consists not in the whole network anymore but only in the layers through which z has not yet been through.

In subsection 3.1, three criteria were derived : performance on source, covariate shift, and conditional shift. We can transpose them to our case to better understand IDR methods. It will appear that they fulfill two out of three.

I **Informativity of the latent representation** : Firstly, extracting discriminative features ensures a low empirical risk of source. $\mathcal{L}_{pred}$ on source is a direct proxy for R_S. In practice, this criterion will never really be problematic. Minimizing the prediction loss in the source domain ensures that the latent representation conserves the information needed to perform the task. The only possible issue might arise from the other terms of the total loss becoming too important, making the importance of the prediction loss negligible. Following the same idea, our neural network architectures will always be supposed to be expressive enough to learn the task for the focus to remain on the distribution shift problem. Regarding the information bottleneck principle, this corresponds to the notion of preserved relevant information.

II **Latent shift** : Secondly, the covariate shift criterion requires that the distributions of the considered representations are similar. To avoid confusion, we will name the notion of shift between latent representation distributions of *latent shift*. The divergence loss $\mathcal{L}_{div}$ minimization assumes that role. For criterion I, we established that minimizing a certain prediction loss was sufficient and did not need further investigation. However, for the latent shift, there are numerous divergences that can be minimized. As seen just above, it leads to the development of several methods to compute and optimize them. Even if all of them are positive metrics converging to zero when the distributions are identical (*we have found an invariant feature*), their behavior when being optimized and their sensitivity to qualitative differences between distributions influence the performance of the method. From section 3.3, we underlined that a covariate shift can be tackled efficiently only if the style component is shifted. On the contrary, if the content component follows different distributions, the latent shift may be irreducible.

III **Encoding-induced conditional shift** : Regarding the last criteria, we saw previously that we could not directly probe it, and we needed the covariate shift assumption, the latter being not too strong. It is not the case anymore. We have no clue how the conditional distributions will be modified by the encoder. It is possible that the covariate shift assumption collapses. We will call this phenomenon *encoding-induced conditional shift*. This uncontrollable behavior will be further explored in section 5. In the latter, a metric to evaluate the risk of this phenomenon will be proposed.

In other words, an IDR method becomes an interesting approach if it discovers an invariant and discriminative feature that preserves equality between conditional distribu-

tions. First, such a feature should exist. Through our interpretation in section 3.3.1, this feature is related to the content component. Thus, as long as they do not face a content shift, the method would be able to reduce the latent shift.

Nevertheless, reducing the latent shift is not yet sufficient, we should stay clear of encoding-induced conditional shift. We will present here a toy example of such a situation that we will more deeply explore in section 5.

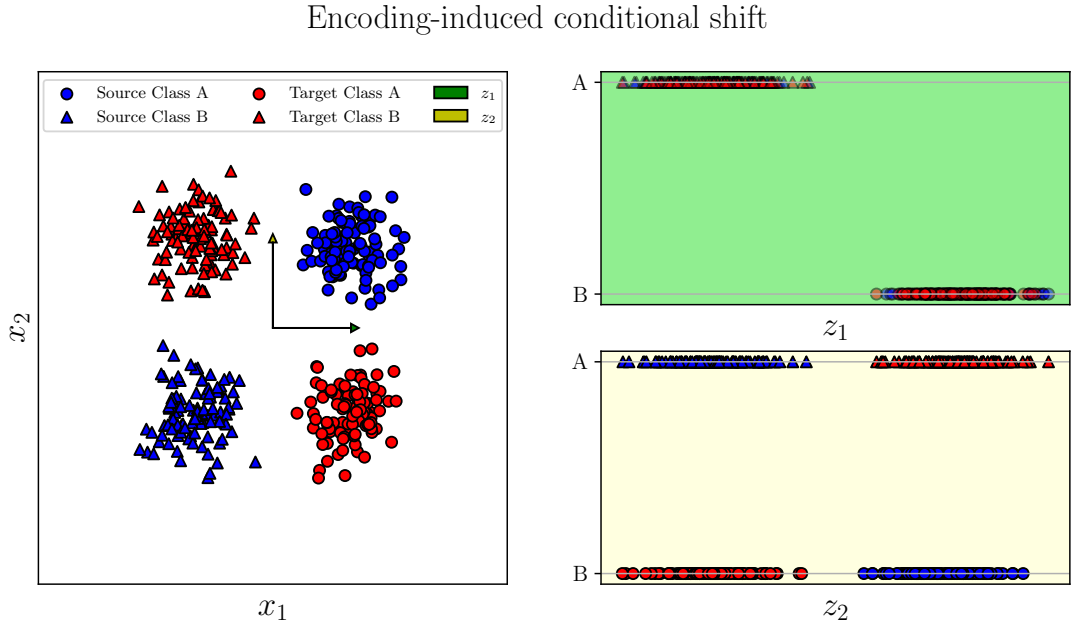


Figure 21: An example of encoding-induced conditional shift. Given the points from the left figure, our encoder could extract *inter alia* the features z_1 or z_2. Indeed, we see from the right figures that source and target features are aligned and that it is easy to classify the points from the source domain. If z_1 is learned, the conditional distributions are equal $\mathcal{S}_{Y|Z} = \mathcal{T}_{Y|Z}$ and the classification will generalize to the target domain. On the contrary, if z_2 is extracted, the conditional distributions are totally different $\mathcal{S}_{Y|Z} \neq \mathcal{T}_{Y|Z}$.

	I	II	III
ERM learning bounds	Informativity of X on Y	Covariate shift	Conditional shift (Absent under covariate shift assumption)
IDR learning bounds	Informativity of Z on Y	Latent shift	Encoding-induced conditional shift
Loss term	$\mathcal{L}_{\text{pred}}$	$\mathcal{L}_{\text{div}}$	None
Failure cause	Unadapted architecture	Content shift	Spurious correlation
Impossibility theorems	First impossibility theorem (Theorem 3.6)		Second impossibility theorem (Theorem 3.7)

Table 4: Parallel between *a priori* learning bounds under distribution shift and their evolution with IDR methods.

5 Impossible domain adaptation

Through the theoretical considerations, the meticulous reader could have noticed that no result has been derived to ensure the feasibility of domain adaptation in general. On the one hand, learning bounds tell us how badly any hypothesis could perform under distribution shifts. In other words, it motivates domain adaptation, finding models significantly more accurate than in this worst case. Also, we saw that these bounds could not be computed by using only available data in the DA setting. For this reason, the covariate shift assumption came up. On the other hand, impossibility theorems demonstrate the necessity of several assumptions (including covariate shift) for DA to be possible in simple cases. Therefore, statistical learning theory does not reassure us about the feasibility of domain adaptation.

Nonetheless, the notions hereabove are derived without considering the structure of DA methods. It would feel unnatural to declare that DA is not always feasible before even taking a single look at how methods try to tackle it. Unfortunately, the theoretical interpretation of them showed us that they may suffer from uncontrollable spurious behaviors. Once again, there is no off-the-shelf guarantee that domain adaptation will solve the distribution shift.

In this section, we will try to better understand and define pathological cases. Our main questions are: When do such situations occur ? And how can we identify them *a priori* ?

An illustrative example :

Heretofore, we have underlined the absence of guarantees. Yet, one could hope for them to be derived one day. To answer this question, let us introduce a perfectly valid example of domain adaptation, which will convince us that it is unsolvable. The latter is represented on figure 22. The source domain is composed of pictures of tigers and horses

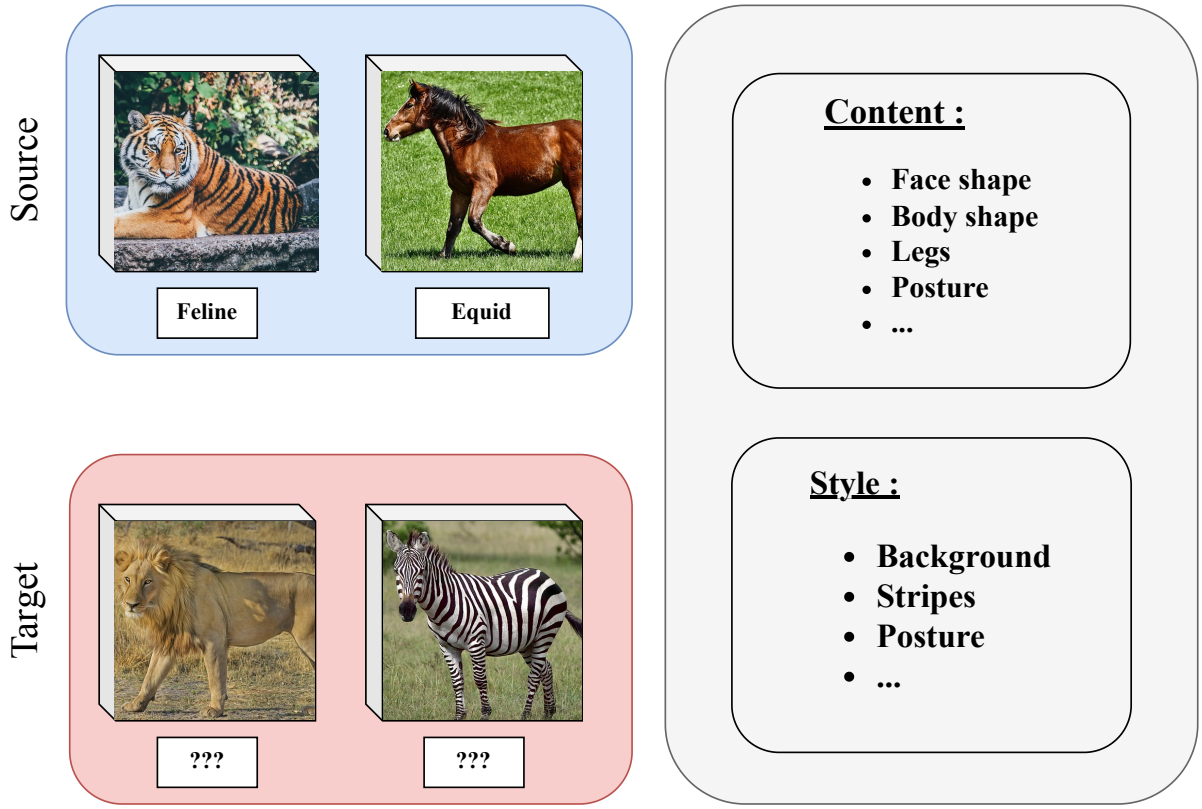


Figure 22: A possibly pathological case of domain adaptation.

and target of lions and zebras. The task is to guess the family of the represented animal (feline or equid). The covariate shift assumption is verified. Indeed, for the same image, the label does not change when switching from source to target domain.

However, one can guess that we cannot ensure that tackling the shift is possible. Let us imagine that one model spuriously deduces that the relevant feature to predict the family is the presence of stripes (the animal is a feline if it is striped, an equid if not). This will be completely exact on source domain. Furthermore, that feature is also present in the target domain and is equally distributed. Notwithstanding, the deduction that a striped animal is a feline is always incorrect now.

5.1 Interpretation through the generative process

Before, we established the notions of style and content of covariates. In DA and under covariate shift, the main goal is to extract the content from the style. What goes wrong in the example above? Firstly, we notice that on each domain, the style component is influenced by the label, $I(y;\varsigma) > 0$. This phenomenon is called spurious correlation, it means that labels are correlated with non-semantic features that could be specific to a given domain and may prevent models from generalizing efficiently. In [Beery et al., 2018], they showed how a cow/camel classifier could be biased by the background of the image (grass or desert), which could seriously harm the out-of-distribution generalization. In spite of that, we dispose of more data than in the classical supervised ML setting as the inputs of the target domain are accessible. This could be used to detect the spurious

correlation and correct it, for example, if the spurious features are not present in the target domain.

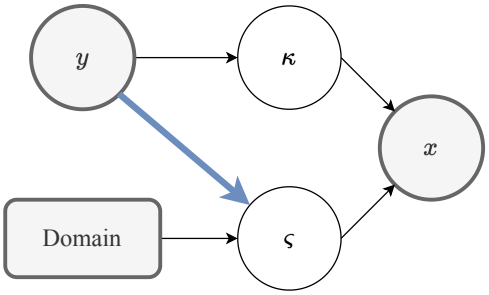


Figure 23: Edge producing spurious correlation in the Bayesian network

Out of these observations, we deduce that DA can fail if there exists a component of the style ς_{sp} such that :

I $I_S(y;\varsigma_{\text{sp}}) > 0$: The spurious component is correlated with the label in the source domain.

II $\mathcal{S}_{\varsigma_{\text{sp}}} = \mathcal{T}_{\varsigma_{\text{sp}}}$: The spurious component is present in both domains and is identically distributed.

In the case exposed in figure 22, the spurious component is the presence of stripes. Also, these conditions do not certify the failure of DA. It only indicates that such a phenomenon is possible. The two conditions listed above are not tractable in a practical setting either. Indeed, one needs access to labels from both domains to distinguish what constitutes style or content. This pitfall of domain adaptation has also been exposed in [Zhao et al., 2019].

5.2 Information bottleneck and spurious correlations

To better understand how such generalization failures occur, let us take a look at the information bottleneck theory. When introducing the latter, we listed the main quantities implied by the IB principle on table 1. How do they behave in such cases ? One should now also notice that these quantities can be differentiated in terms of domain. For example, a feature can be relevant in one domain $I_S(Y;Z) \gg 0$ and irrelevant in the other $I_T(Y;Z) \approx 0$.

To discover how the IB principle can help us define pathological cases, let us sum up what happens in the zebra/tiger case from figure 22. By extracting the presence of stripes, the encoder fulfills the goal that has been addressed to it : finding a feature that is discriminative on source $I_S(Y;Z) \gg 0$ and present on both domains. In spite of that, there were other features fulfilling the same conditions that were completely discarded by the encoder. For example, the presence of hoofs would have worked as well and on top of that, it permitted good generalization. The main problem is that, given complete learning pairs on source and only inputs on target, it is not possible to distinguish these two kinds of features. The only thing we can prescribe for the encoding is retaining a collection of

features that all seem relevant. This quantity is directly related to the residual mutual information $I(X;Y|Z)$.

More formally, we argue that a null residual information is a sufficient condition for the absence of a conditional shift under the covariate shift assumption. Interestingly, we note that in [Johansson et al., 2019], the authors require the invertibility of the encoder to ensure the absence of a conditional shift. This is a strong assumption that is difficult to reach in practice while minimizing the latent shift. Invertibility is a sufficient condition for the absence of residual information.

Proposition 5.1. *(Proof in appendix C)*
Invertibility of the encoder implies a null residual information.

Theorem 5.1. *(Proof in appendix C)*
The covariate shift assumption is sufficient for the absence of a conditional shift when the encoding procedure yields null residual information. If $I(X;Y|Z) = 0$ *and* $\mathcal{S}_{Y|X} = \mathcal{T}_{Y|X}$ *then* $\mathcal{S}_{Y|Z} = \mathcal{T}_{Y|Z}$.

Complementary encoding When exposing our results, we will try to observe the residual information for pathological and regular cases. Here, we present the idea of complementary encoding that will be used to evaluate the residual information.

So far, we want to better understand the interplay between the input, the output and the hidden representation in between. As the encoder generating the latent representation is in general not invertible, we may be looking for the "missing part" of x in latent space. In other words, we can try to develop a complementary encoder yielding z' such that $x = \mathsf{DEC}(z, z'; \varphi)$ with DEC a mapping being used as a decoder. In this setting, z remains the latent vector discovered by the considered DA model.

In practice, we will introduce two more neural networks in the picture, a complementary encoder ENC' and a decoder DEC.

$$\min_{\vartheta,\varphi} \mathbb{E}_{x\sim\mathcal{D}_X} ||\mathsf{DEC}(\mathsf{ENC}(x;\theta), \mathsf{ENC}'(x;\vartheta);\varphi) - x||^2$$

The distribution $\mathcal{D}_X$ is the joint distribution of the source and target domains. The complementary networks annexing themselves to the root architecture of IDR methods yield the structure represented in figure 24.

With the complementary encoding step being done, we can re-express the residual information, which was our primary goal.

$$I(X;Y|Z) = I(Z, Z';Y|Z) = I(Z';Y|Z)$$

We finally conclude that an encoding-induced conditional shift may arise when, for a fixed z, the variable z' is still informative regarding the label prediction. This is the case when the residual information is non-null.

A toy example : Let us observe how these new quantities behave on a toy example of an impossible case as introduced at the end of section 4 on figure 21. The encoder will be made of a simple projection along a given direction of angle θ. This can be seen as a simplification of a neural network with a two-by-one fully-connected layer. In this setting, we have two possible θ minimizing the latent shift. However, for one of them a

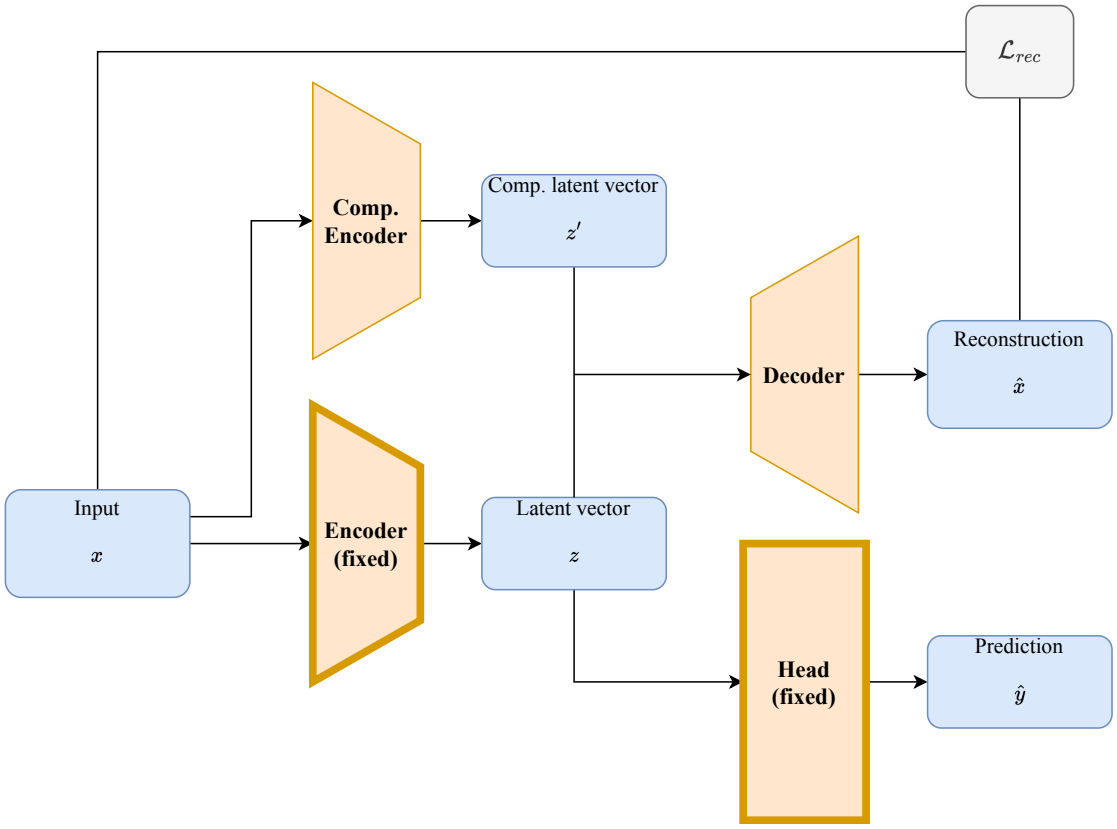
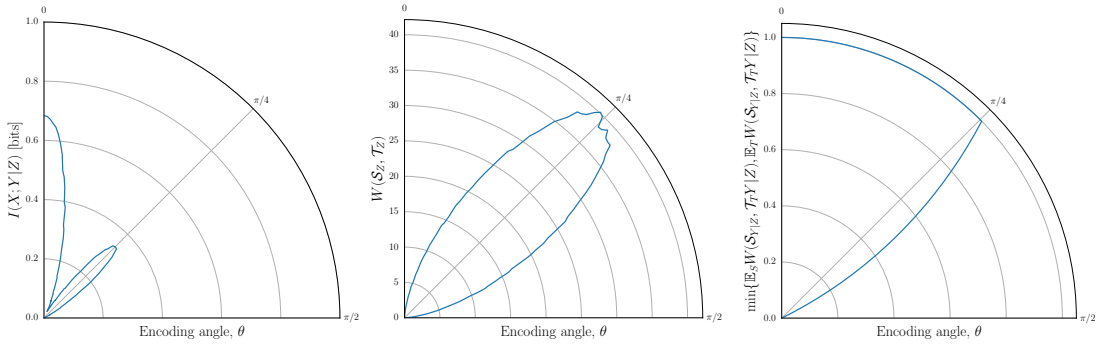


Figure 24: Modification of the IDR method structure to add a complementary encoder.

conditional shift is induced. On figure 25, we can observe that residual information is null for the ideal case without conditional shift and positive for the other.



(a) (b) (c)

Figure 25: We take the example made of four gaussian blobs, the angle θ represents the orientation of the projection plane of the encoder (a) Residual information, it reaches 0 only for $\theta = \pi/2$ (b) Wasserstein distance between latent distributions, as already seen on figure 21, the minima are at $\theta = 0$ and $\theta = \pi/2$, an IDR methods should therefore one of these two values (c) Conditional shift metric, the only angle that does not suffer from encoding-induced conditional shift is $\theta = \pi/2$.

6 Numerical experiments of Unsupervised Domain Adaptation

In section 4, we selected a group of methods addressing the problem of UDA from different perspectives. Then, we established from high-level considerations and theoretical aspects that there exist problematic configurations in which DA cannot guarantee any results. Now, it is time to practically observe the behavior of these methods. Firstly, following the same philosophy as for selecting methods, we will present different distribution shifts and tasks to obtain representative results. Instead of repeating similar conclusions or spotting differences, our idea is to explore some theoretical behaviors through each of our cases. Secondly, we will introduce an artificial pathological case and observe the consequences. We will also present results on inverse problems to better understand the behavior of the methods in such conditions.

Our philosophy for observing aimed behavior is to introduce controllable distribution shifts, *i.e.* a tuning knob to control the distance between source and target distributions. Therefore, we will not only always consider a unique target domain but a set of them such that the domains are gradually shifted. Yet, the *direction* of the shift (style or content divergence, spurious correlation, ...) will depend on the example. Therein, we differ from most of the literature and *a fortiori* from the traditional surveys that focus on a single shift (for example MNIST/MNIST-M or MNIST/SVHN which is a very common benchmark example).

Regarding implementation details and hyperparameters, the whole procedure is described in appendix D. Briefly, architecture and hyperparameterization are tuned such that they yield significant results on each example, *e.g.*(convolutional encoder with dropout for images, dense encoder for tabular data, ...) without over-complicating the training phase and wasting computational resources. The tuning is done using a classical cross-validation methodology. For the benchmark to be fair, the same architecture is used for the encoder and head for all methods across each example. When exclusive neural networks are required, *e.g.* critic in WDGRL or domain discriminator in DANN, the architecture has been chosen to be as similar as possible.

Experiment	Instances	Tasks	Content	Style	Shift parameter
Noisy digit classification	Handwritten digits	Classification	Digit	Added shapes	Number of shapes per digit
Spuriously correlated digit classification	Handwritten digits	Classification	Digit	Randomly oriented stripes	Correlation between digit and stripes orientation
Spuriously correlated and noisy digit classification	Handwritten digits	Classification	Digit	Added shapes and randomly oriented stripes	None
Deblurring	Blurred images	Image reconstruction	Sharp image	Blur orientation	None
Gravimetry	Gravific field	Image reconstruction	Density field	None	Shapes distribution

Table 5: Summary of experiments

6.1 Noisy digit classification

This first example demonstrates the capacity of IDR methods to tackle a convenient distribution shift for classification. It can be seen as a proof of concept for the different methods in section 4. By convenient shift, we mean a covariate shift due to style shift and without spurious correlations. From the analysis that we have done here above, IDR methods should be able to solve the distribution shift. As it is the first example that is presented, we will use it to illustrate the general behavior of the methods.

6.1.1 Dataset and shift

Our experiment is based on the famous Mixed National Institute of Standards and Technology (MNIST) dataset of handwritten digits [Deng, 2012]. The source domain is made of the original dataset in which we want to identify the digit between 0 and 9 from the image. For the target domain, we add noise to the image while keeping the label unchanged. Adding merely white noise may be too simple, and the network may learn to ignore it without any modification. Therefore, we generate noise by adding geometric shapes (circles, ellipses, squares, and triangles) to the image of random sizes and shapes. The shift is controlled by the number of shapes added. A preview of this is shown in figure 26.

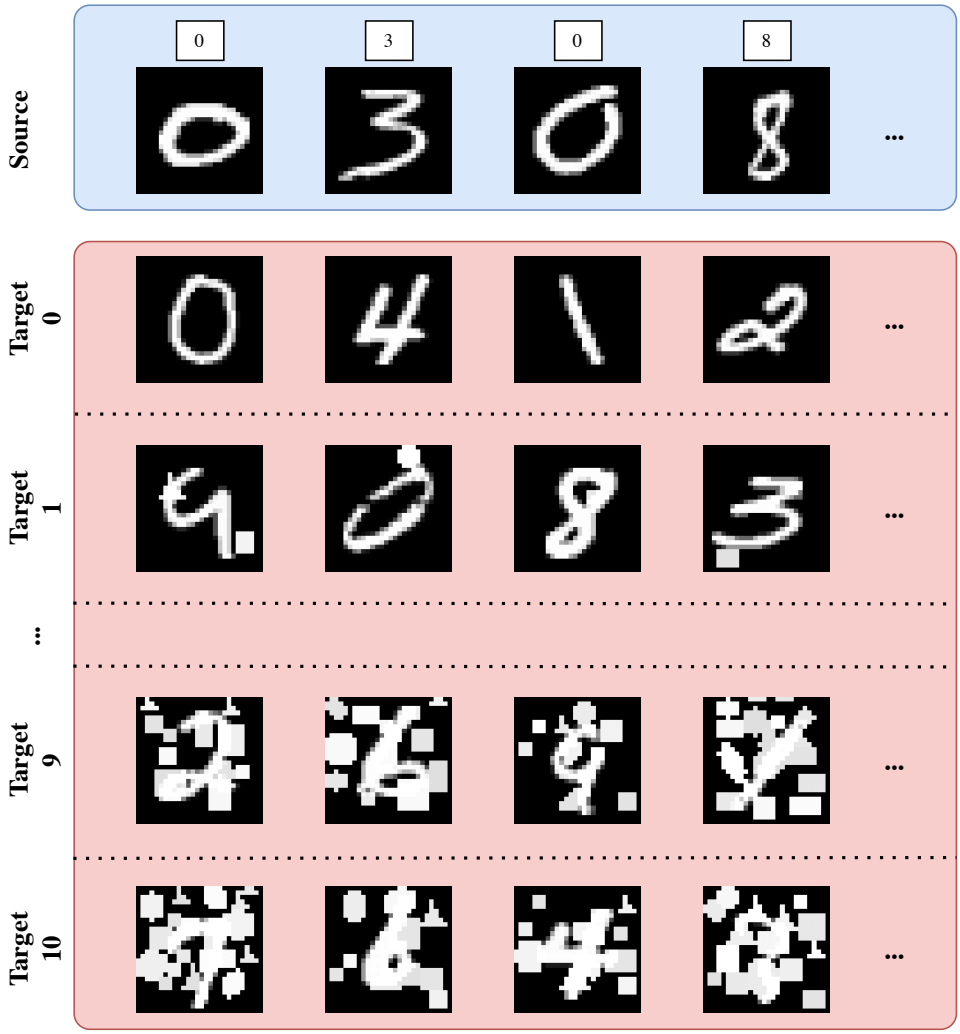


Figure 26: Preview of the considered dataset and shift for the noisy digit classification domain adaptation.

Here, we clearly face a shift due to the style component. Indeed, the content is contained within the digit shape, and the added noise is totally irrelevant to the classification task. Also, there is no spurious correlation, as the noise is added independently of the digit.

6.1.2 Results

In theoretical considerations, we established that IDR methods should work by fulfilling three criteria. These will be our guidelines for interpreting the results.

Obviously, a necessary starting point is performing well in the source domain. We saw in section 3.1 that it theoretically plays a major role in the generalization abilities of the model. From a more practical point of view, it validates our basic learning strategy, *i.e.* layers disposition (type, deepness, size, ...) and training hyperparameters (number of epochs, choice of optimizer, learning rates, ...). This can be verified (independently of the shift, as we consider only the source domain) in figure 27. Indeed, in all cases, the classification accuracy on an unseen test set is around 99% (for a ten-class classification

task). As expected, the train accuracy is higher than the test accuracy.

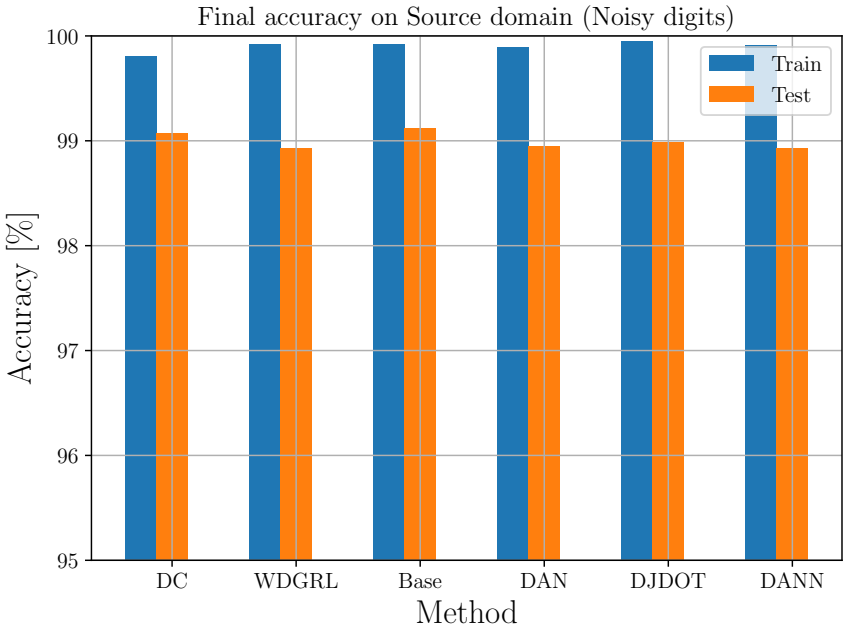


Figure 27: Accuracy on source domain at the end of the training procedure.

Now that we are sure that the architecture is able to learn the task in the source domain, let us take a look at a more challenging aspect, the latent shift. Our goal is to obtain a latent representation that encodes features that are invariant between source and target domains. One way to assess that is by using the Wasserstein distance in latent space. Our intuitive idea would be that all the divergences are equivalent *i.e.* they are all proxies of each other. Consequently, the Wasserstein distance should be smaller when using UDA if we follow this idea.

This behavior can be seen on figure 29. If we observe the distances as such we do not directly observe a clear pattern. Indeed, the Wasserstein distance is not directly smaller for DA methods than for the baseline. A possible explanation is that the heads of the different models may be differently sensitive to distance in latent space. This echoes the presence of the Lipschitz constant of the head in the learning bound (Theorem 3.4). When scaling the Wasserstein distance by the Lipschitz constant of the head, the pattern becomes clearer. We see that the scaled distances converge to similar values for all IDR methods and are smaller than the baseline.

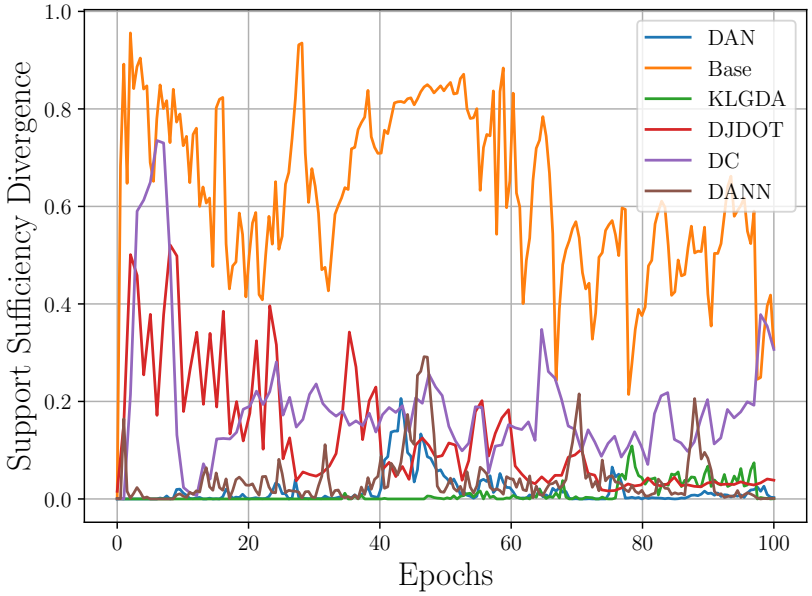


Figure 28: Evolution of the support sufficiency divergence($\epsilon = \frac{1}{10^3}$) between the latent distributions of source and target domains for the different methods. The shift parameter is set to 10.

Yet, we see that for DANN, it reaches a lower value than baseline but higher than other methods. However, it does not perform worse than the other methods. We think that this is due to the support sensitivity of the divergence used for DANN. Jensen-Shannon is a support sensitive metric. Also, the domain discriminator is probably rapidly confused once the supports overlap. This is a possible explanation for the higher Wasserstein distance. If we compute the support sufficiency divergence[3] (Definition 2.11), we see that DANN on figure 28 has one of the lowest values. This may confirm our intuition that DANN aligns the supports rather than the distributions. The fact that it does not affect the performance may be a manifestation of the claim of [Johansson et al., 2019], namely, that the performance under distribution shifts is rather affected by support overlap than divergence between distributions.

[3]The latter is estimated by Gaussian Kernel Density Estimation

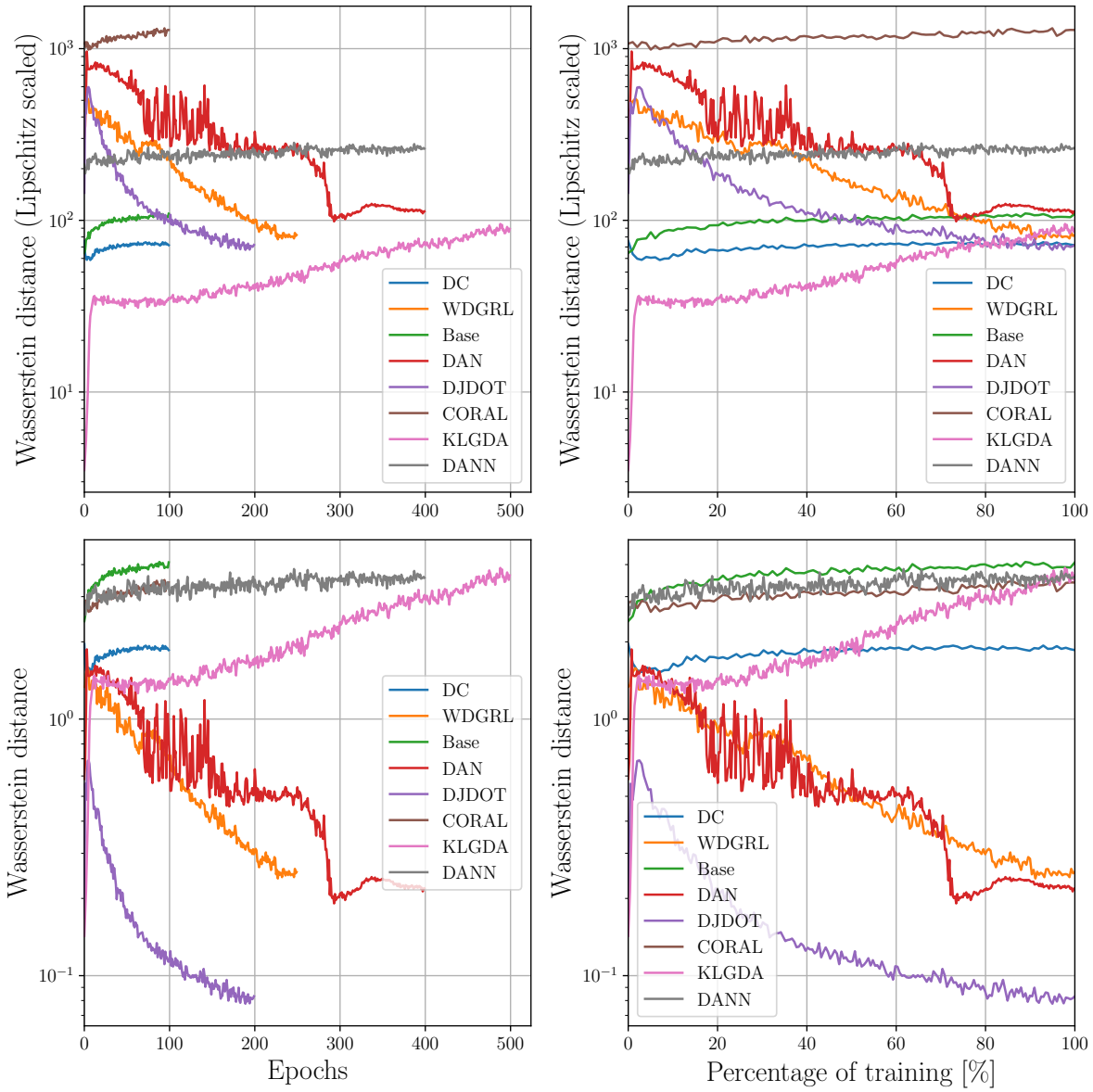


Figure 29: Evolution of the Wasserstein distance in latent space for the different methods during the training process. (Right) Normalized by the number of epochs. (Top) Scaled by the Lipschitz constant. The shift parameter is set to 5.

On figure 30, one can observe the delicate balance between invariance and discriminative power. When the divergence coefficient is null, the model performs almost perfectly on source. It has found discriminative features on the source domain. However, they are not invariant as we can see from the poor target accuracy. When increasing the coefficient, latent representations grow in invariance, as the divergence decreases. When chosen too high, the features are indeed invariant but not discriminative anymore as the accuracy on source and target drops. With the Goldilocks principle, we can find a trade-off value that promotes both aspects and, therefore, maximizes the accuracy in the target domain.

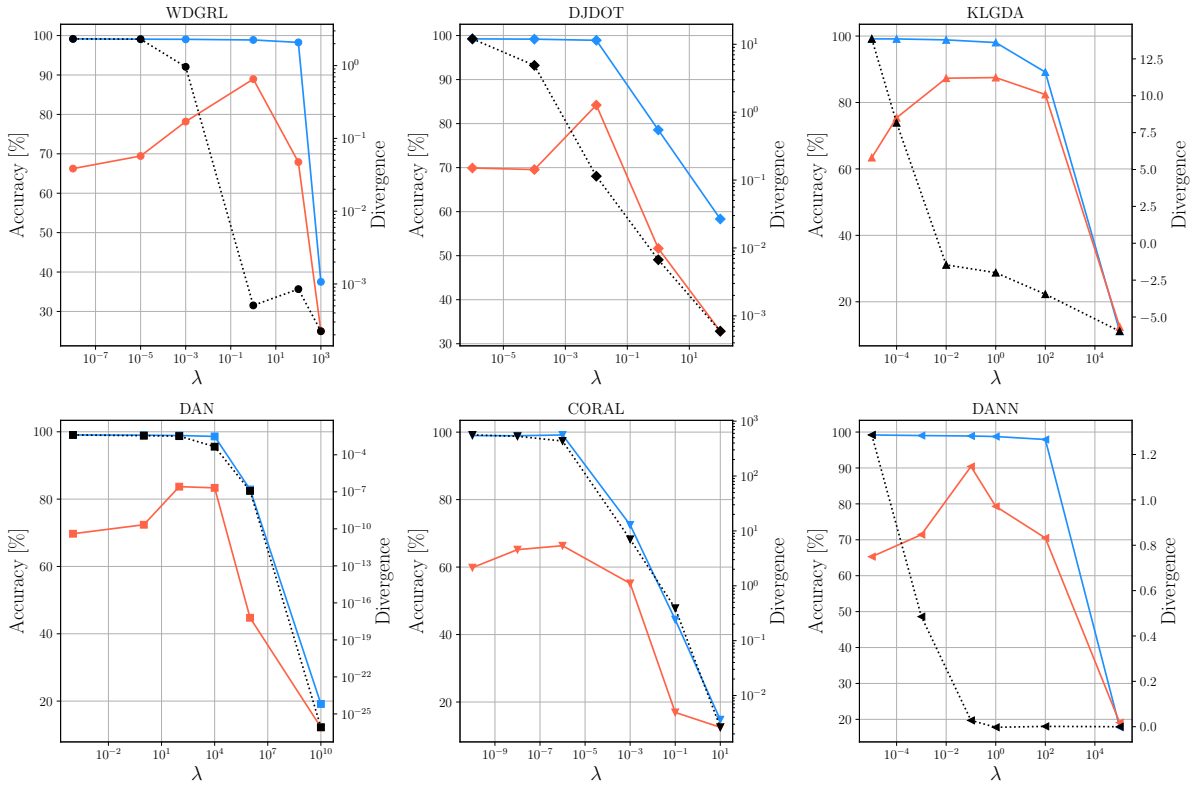
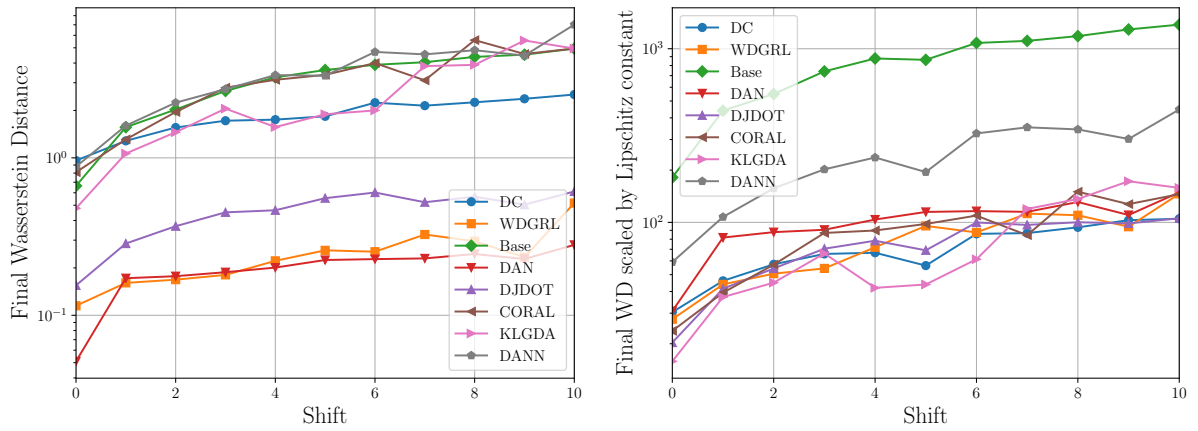


Figure 30: Influence of divergence coefficient for each IDR method. In blue, accuracy on source; in red, accuracy on target domain; black dotted lines represent the evolution of the associated divergence. The shift parameter is set to 5.



(a) (b)

Figure 31: (a) Evolution of Wasserstein distance between the latent distributions of source and target domains for the different shifts. (b) Evolution of Wasserstein distance between the latent distributions of source and target domains scaled by the Lipschitz constant for the different shifts.

In this setting of purely style shift, we suppose and observe that the encoder is able

to extract meaningful features on both domains. However, as the shift increases, these invariant features become more and more difficult to extract. These would result in the latent shift growing with the shift. This can be observed in figure 31.

The dataset has been designed without a spurious correlation between style and content. Consequently, it should not suffer from encoding-induced conditional shift. This can be confirmed by observing the residual information on figure 38a.

Now, let us observe some specific behaviors of the different methods we use. Some of them require additional procedures, and we will observe the effect of the latter on global training.

For Domain Adaptation Network, the kernel inducing the MMD is modified along epochs to fine-tune the sensitivity of the discrepancy. On figure 32, we observe how the weights of the different kernels evolve. Their evolution compared to the Wasserstein meets our expectations. Indeed, as they change, the Wasserstein distance decreases, which emphasizes the beneficial effect of tuning the weights (*i.e.* compared to Deep Domain Confusion). Here, we consider radial basis functions with different bandwidths. And we can see that, at the end, the kernel with the smallest bandwidth grows in importance. This is probably due to the fact that source and target distributions are becoming closer in the latent space, and they can discriminate only on small scales.

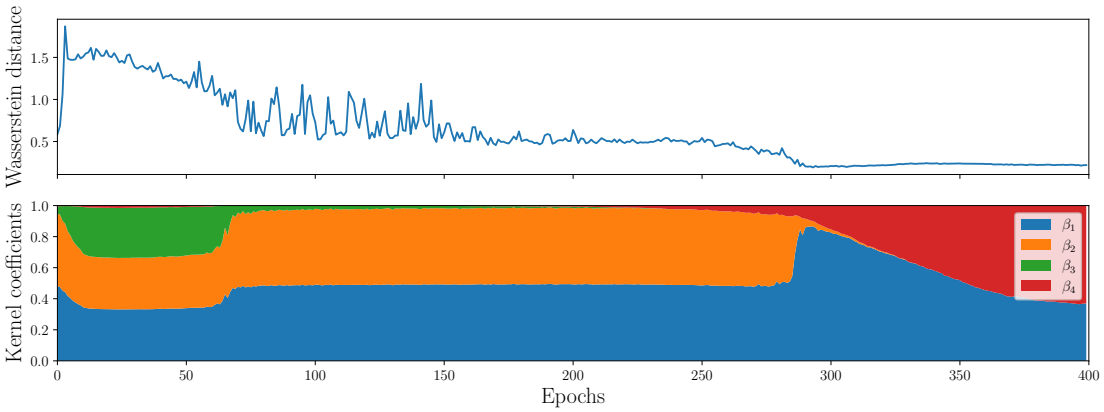


Figure 32: Evolution of the Wasserstein distance (top) for DAN. Evolution of the kernel weights for the MK-MMD in DAN (bottom). The shift parameter is set to 5.

For DANN, an additional domain discriminator has to adversarially promote invariance. On figure 33, one can see that the accuracy on the target domain increases rapidly as the confusion of the discriminator increases at the same time. It finally reaches a plateau of 50 % accuracy which corresponds to its maximal confusion.

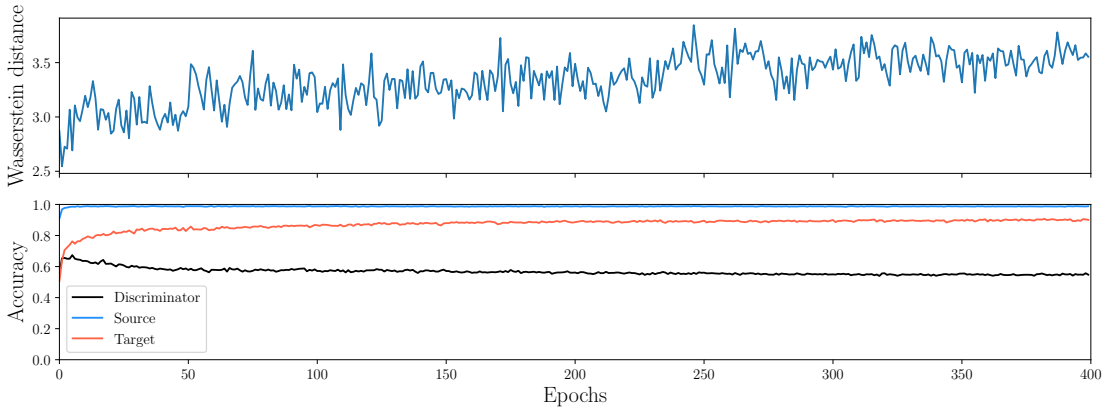


Figure 33: Evolution of the Wasserstein distance (top) for DANN. Evolution of the accuracies of the head and the discriminator (bottom). The shift parameter is set to 5.

When introducing Deep Joint Distribution Optimal Transport, our hope was that the OT solver would return a label-conservative transport plan. For example, all the ones from the target domain should be associated with ones from the source. On figure 34, the label-wise transport plan evolves toward this ideal behavior (which corresponds to the diagonal). This means the unlabeled samples from the source domain receive mostly the correct label through the application of the transport plan.

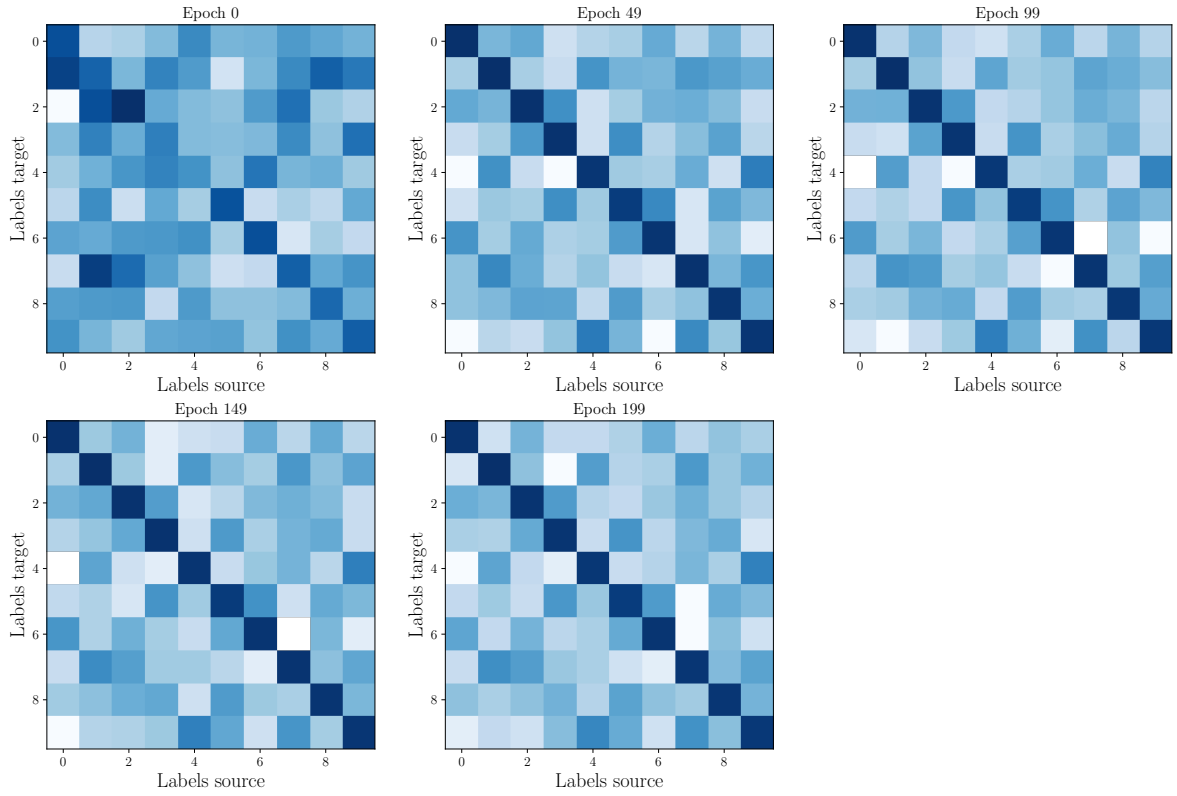


Figure 34: Evolution of the transport map with respect to the labels in source and target domains for DeepJDOT (log-scale color map). The shift parameter is set to 5.

We now focus on Wasserstein Distance Guided Representation Learning. For this method, we try to compute the Wasserstein distance as for DJDOT. However, for DJDOT, it relies on the primal formulation of the OT problem, and we use an OT solver to compute the distance. In WDGRL, we rely on the Kantorovitch-Rubinstein duality (which can be seen as the dual form of the problem). We now compare the evolution of the two approximations on figure 35.

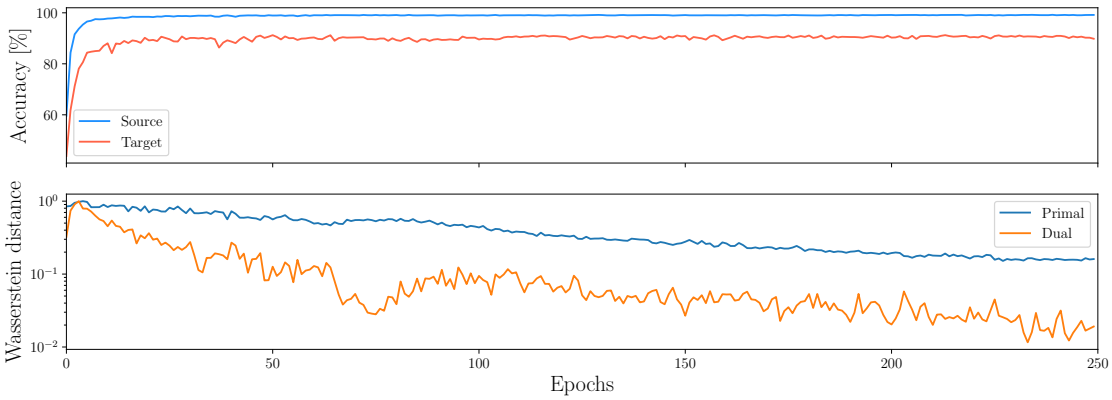


Figure 35: Evolution of the accuracy for WDGRL. (top).Evolution of the two estimations of the Wasserstein distance. The shift parameter is set to 5.

Through this example, we have mainly been able to observe how DA methods reduce the latent shift. We saw that the smallest gain in accuracy is obtained by CORAL. This is not surprising, as it leverages a divergence that lacks expressivity. On this example, the best average performance is obtained by Wasserstein Distance Guided Representation Learning. Its gain is even more important as the shift increases. All the different accuracy values are shown on figure 36. The efficiency of domain adaptation methods is shown by all the accuracies being higher than the baseline. As expected by theory, the accuracy decreases with the amplitude of the shift.

An interesting consideration is the computational cost and complexity of each method. Even if they share comparable architecture for the encoder and the head, computing the divergence can be more or less complex. It would be too complicated to compare them properly (some divergences are computed iteratively, others directly, some can be parallelized or not, *etc.*), but we will consider high-level aspects. We underline that WDGRL and DANN require training of neural network dedicated to divergence computation, there are also the two best performing approaches in this example. The computational cost can be scaled by the architecture of the discriminator or critic. Then come methods that involve solving an optimization problem at each iteration, *i.e.* DeepJDOT and DAN. The optimization problem for DAN is simpler than for DeepDJOT as it is a convex problem. Finally, KLDGA, CORAL and DDC use divergence metrics that can be computed directly from the data. In practice, KLGDA is the most computationally expensive because of the approximation it requires. Consequently, one could notice that DDC performs very well regarding its computational cost.

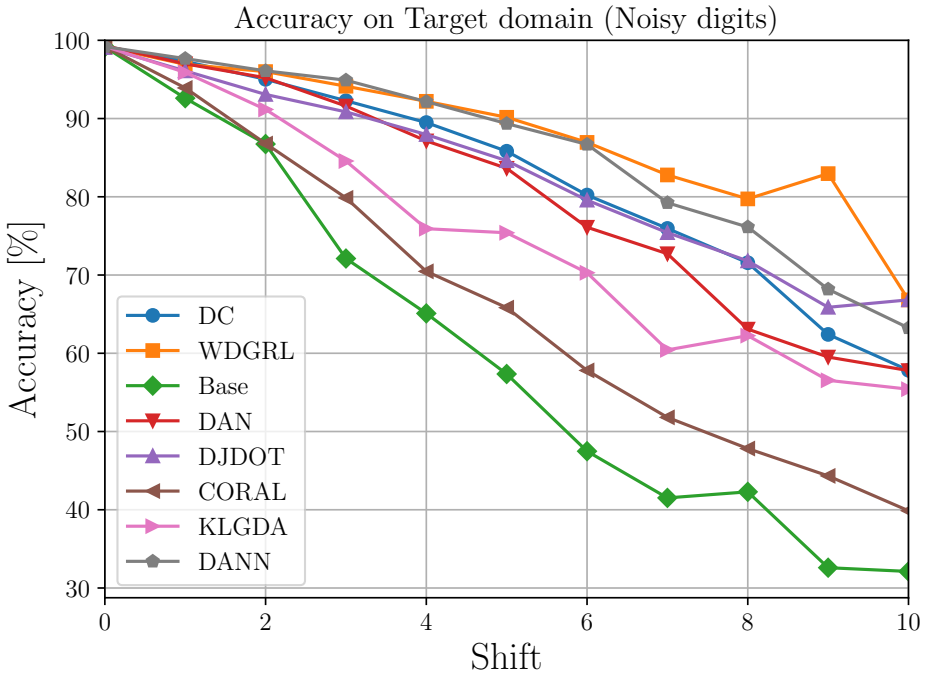


Figure 36: Accuracy on target domains for the different methods and shifts.

6.2 Spuriously correlated digit classification

In the literature and in the previous sections, impossible DA was discussed in theory and linked with other frameworks. Here, we will illustrate it with a controllable toy example of spurious correlation. Experiments will show that DA cannot tackle the distribution shift.

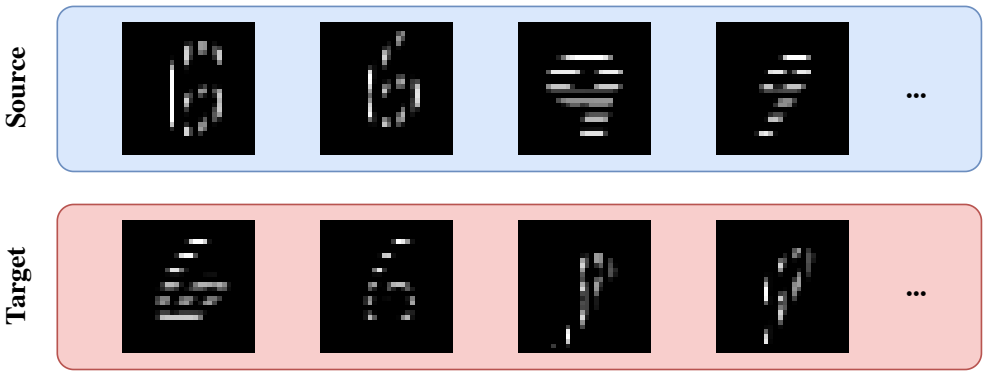


Figure 37: Preview of the considered dataset and shift for the spurious correlation example. This corresponds to the most extreme shift, other shifts are described in table 6.

6.2.1 Dataset and shift

To keep the problem simple, let us consider a binary digit classification based on the MNIST dataset. Now, the model must only identify handwritten six and nine (as they

share common shape elements, the task is not too straightforward). As for the first example, we will introduce noise to the image. This time, it will be composed of randomly oriented stripes. The shift is controlled by the correlation between the orientation of the stripes and the digit. A preview of this is shown on figure **??**.

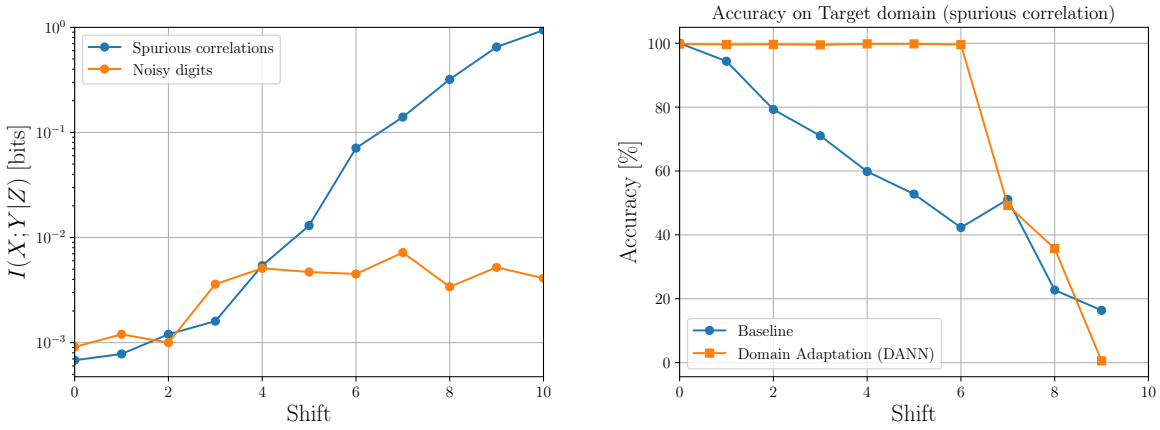
The challenging aspect here is that the stripes are not the real discriminative feature. A robust learner should rely on the digit shape to classify the image. Here, the UDA methods cannot use the unlabeled target samples to enhance their feature extraction.

Source domain		Target domain	
6	**9**	**6**	**9**
$\pi/2 + \mathcal{U}(-\alpha, 0)$	$\mathcal{U}(0, \alpha)$	$\mathcal{U}(0, \alpha)$	$\pi/2 + \mathcal{U}(-\alpha, 0)$

Table 6: Stripes orientation for the spuriously correlated example. The parameter α decreases linearly from $\pi/2$ to 0 depending on the shift parameter.

6.2.2 Results

The dataset disposes of a spurious correlation between style and content. Therefore, we cannot ensure that the DA methods will not suffer from encoding-induced conditional shift. This would cause the DA methods to fail by not outperforming the baseline. We can see on figure 38b how domain adaptation fails as the spurious correlation strengthens. The accuracy finally drops to the baseline level.

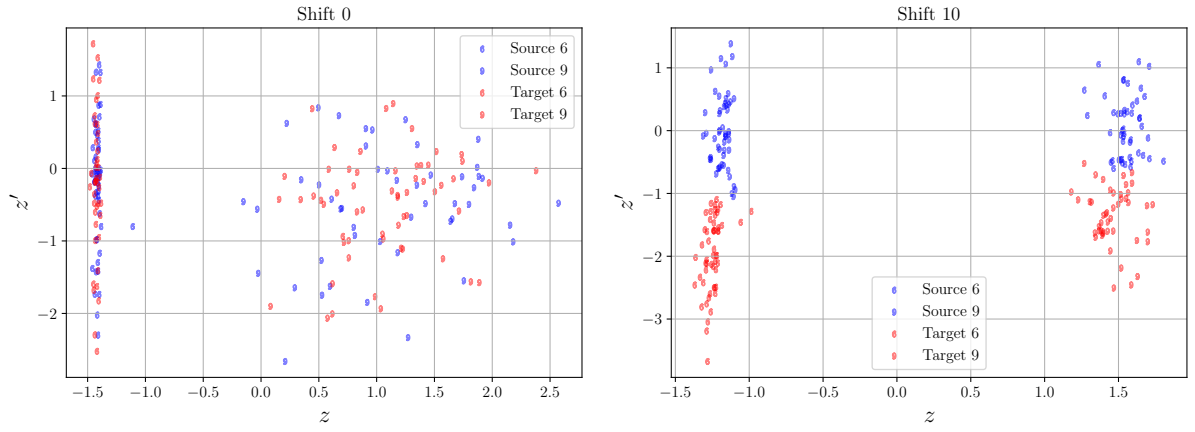


(a) (b)

Figure 38: (a) Evolution of the residual information estimated using complementary encoding for the ideal (without spurious correlations) and pathological shifts. (b) Accuracy on the target domain for the baseline and one representative method (Domain Adversarial Neural Network).

Previously, we stated that it was possible to measure the risk of a conditional shift by measuring the residual information. On figure 38a, the residual information increases with the amplitude of the shift. It finally even reaches one bit, which is the maximum value for a binary classification task.

Using complement encoding, we can observe how the augmented latent space is structured. On figure 41, we see that when the style is uncorrelated with the content on both domains, z' is totally irrelevant to the classification. On the contrary, when the style is correlated with the content, for a given z, z' contains information about the label. This confirms more visually the increase in residual information.



(a) (b)

Figure 39: (a) Latent space distributions using complementary encoding without shift. (b) Latent space distributions using complementary encoding with shift. The shift parameter is set to 10.

Here, we have been able to numerically confirm the impossibility for IDR methods to tackle the distribution shift when a spurious correlation is present. The estimation of residual information using complementary encoding has been a good indicator of the risk of encoding-induced conditional shift.

6.3 Spuriously correlated and noisy digit classification

We will now mix the two previous examples to answer the following : Is the DA feasible if there is a spurious correlation only on the source domain ? Following our considerations in section 5 methods should not fail. It is also an interesting experiment that will emphasize the capacity of IDR methods to extract invariant features.

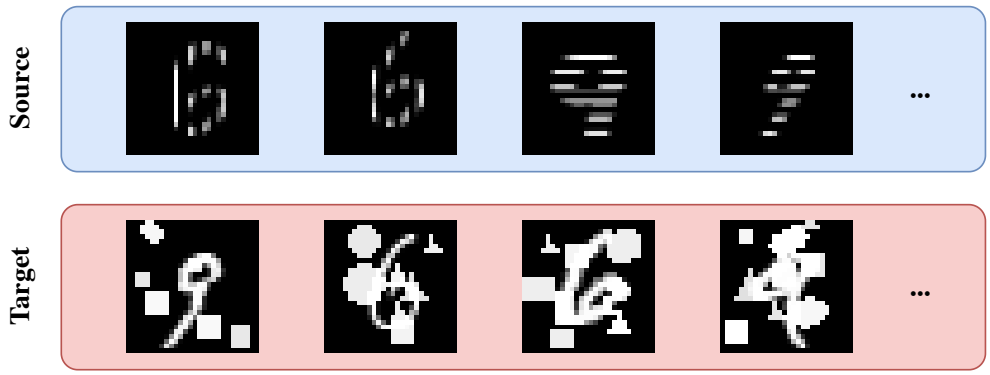


Figure 40: Preview of the data of the third experiment.

6.3.1 Dataset and shift

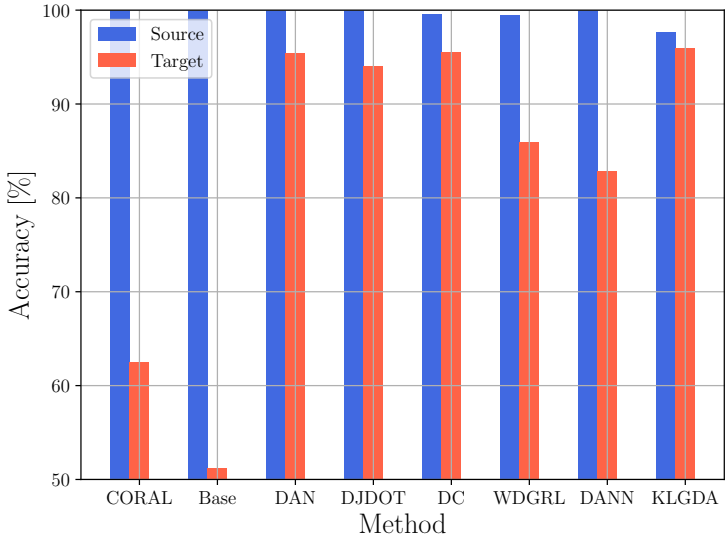
Once again, the task consists of a two-digit binary classification (with six and nine). Here, we will not consider a tunable shift. The source domain corresponds to the extreme shift of the previous example, namely that the six are horizontally striped and the nine are vertically striped. The target domain is similar to what we used for the first experiment, *i.e.* the digits are corrupted by geometric shapes. As we said, the results of this shift will clearly demonstrate the capacity to extract invariant features. Indeed, there are two discriminative features in the source domain : the stripe orientation and the digit shape. However, only the digit shape is invariant and should be extracted by the model. In that sense, this is a more challenging shift than the first one.

6.3.2 Results

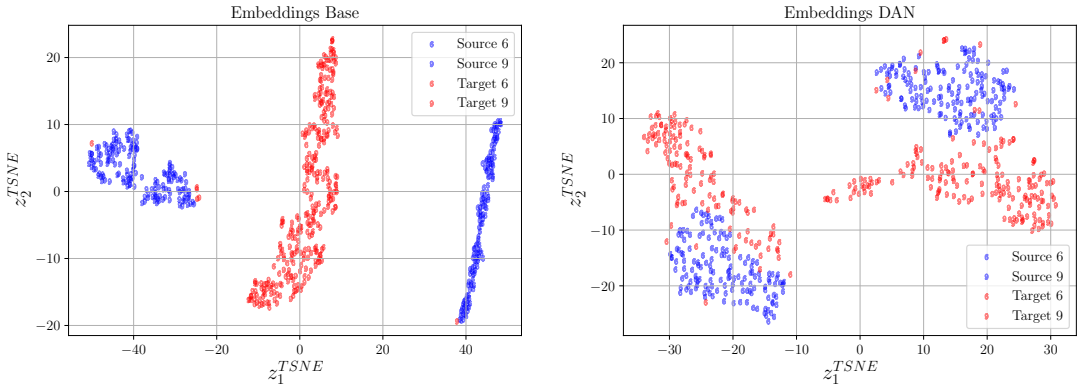
On figure 41a, one can see the success of DA methods on this task. Indeed, the baseline approach is unable to classify the digit on the target domain (accuracy around 50%). On the contrary, all IDR methods are able to beat that score. This means that they can extract the invariant feature (the digit shape) even though there is a spurious correlation on the source domain. The lowest scoring DA method appears to be DeepCORAL. This result is not surprising as it was also the poorest IDR method in the first experiment. We consider that this is due to the weakness of its divergence.

Comparing the second and third experiments confirms our claim that the existence of a spurious correlation in only source domain does not prevent DA methods from succeeding. Encoding-induced conditional shift appears as soon as there is a spurious correlation on the target domain.

From the representation of the latent space on figures 41b and 41c, it appears clearly that the baseline model manages to classify the source domain correctly while being unable to extract relevant features on target. But for UDA, the model is able to extract the invariant feature by correctly clustering the latent vectors and associating the source and target clusters. Here, we considered DAN for the visualization, but similar results are obtained for the other methods.



(a)



(b) (c)

Figure 41: (a) Accuracy on source domain and target domain at the end of the training procedure for the third experiment. (b) Latent representations distributions for the baseline model (visualized with t-SNE). (c) Latent representation distributions for the best performing DA method, domain adaptation network (visualized with t-SNE).

This example confirms that the methods can extract invariant features that would not have been detected by a classical model. This clearly shows the link between UDA and the generalization abilities of neural networks.

6.4 Deblurring and operator shift

In most of the literature, UDA is tested on image classification tasks, (leading to the emergence of many classification-specific UDA methods). Of course, there are examples of regression or other computer vision tasks (segmentation, detection, pose estimation, ...). Regarding inverse problems, the literature is not as rich, and we want to present here the power of UDA when leveraged for these tasks. Indeed, one can establish a parallel between the notions of style and content and the notions of operator and data in IP. We will illustrate this with a deblurring task.

6.4.1 Dataset and shift

As explained, in the section about covariate shift 3.3, a style shift can be obtained by shifting the observation operator. Here we consider two different idealized motion blur operators. The source domain is affected by a horizontal motion blur and the target domain by a vertical motion blur as shown in figure 42.

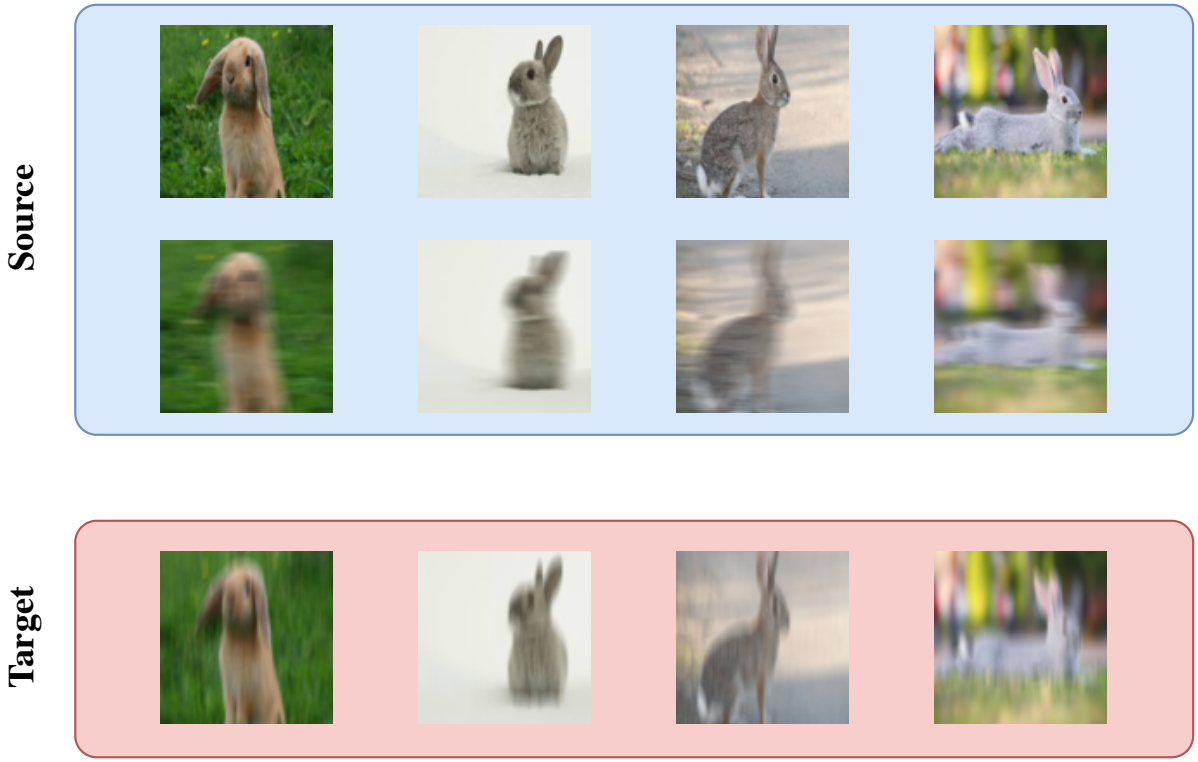


Figure 42: Preview of the considered dataset and shift for the deblurring domain adaptation.

The objective of our model will be to restore a sharp image from a blurred one. Thanks to the labeled source domain, the model can extract a prior on the sharp image distribution. Then, the objective of the UDA method is to use the unlabeled target data to recognize the target forward operator and adapt the encoder to it.

6.4.2 Results

For this task, we chose the U-Net architecture as it is particularly suited for deblurring tasks and computational imaging in general. It fits in the IDR framework if we consider the latent representation as the union of the different representations sent to the decoder.

If we solely use the U-Net to deblur on source domain, then feed it with target data, important artifacts will appear on the reconstruction, as it has never encountered this distribution before. This can be seen on figure 43. As for the previous examples, this constitutes our baseline results. The goal of UDA will be to reduce these artifacts by incorporating unlabeled target data.

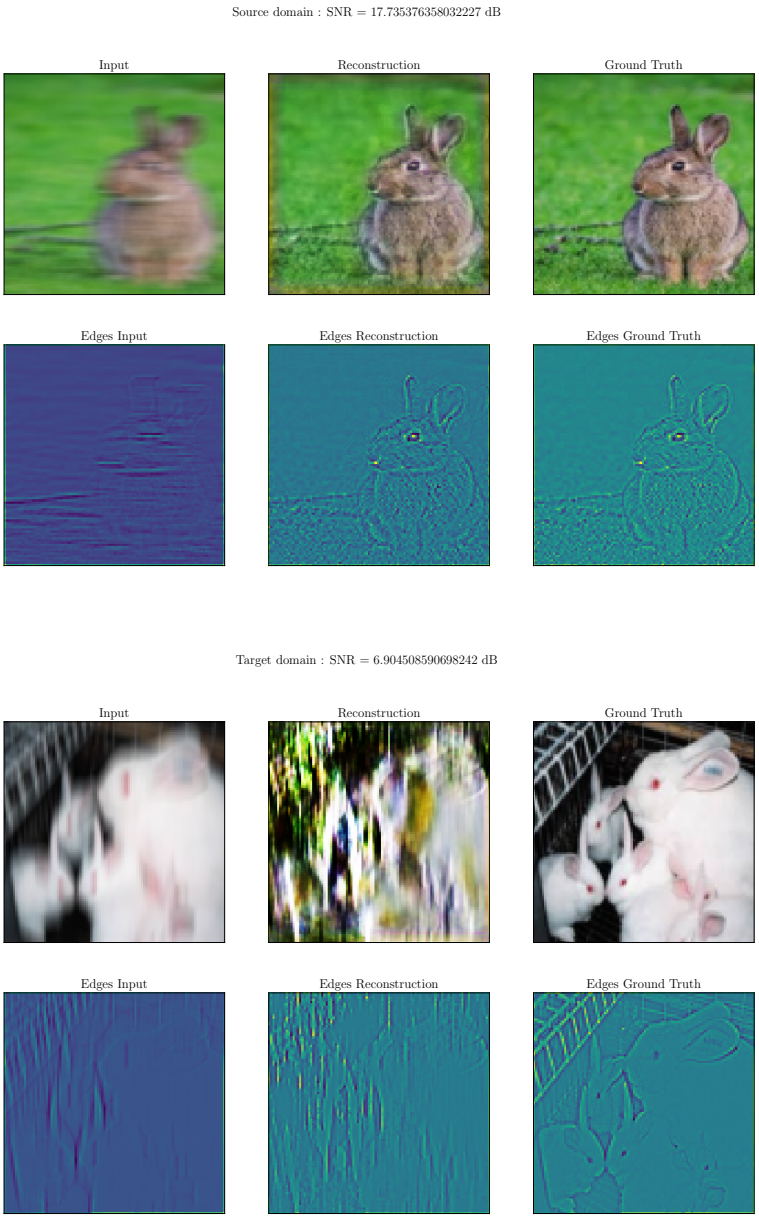


Figure 43: Baseline results for the deblurring example. Reconstruction of a source image (top) and a target image (bottom) using the U-Net without adaptation.

A discriminator could probably distinguish between source and target baseline reconstructed images. Introducing it in the architecture and training it adversarially should help the encoder extract features that are invariant to the operator shift. On figure 44, we observe the efficiency of DA. Indeed, by increasing the need for invariance in the loss expression (by increasing the divergence coefficient), test loss on the target domain decreases. As for the other examples, we see a trade-off between invariance and discriminative power. When the invariance coefficient becomes too high, invariance is indeed achieved, but the reconstruction quality drops on both domains.

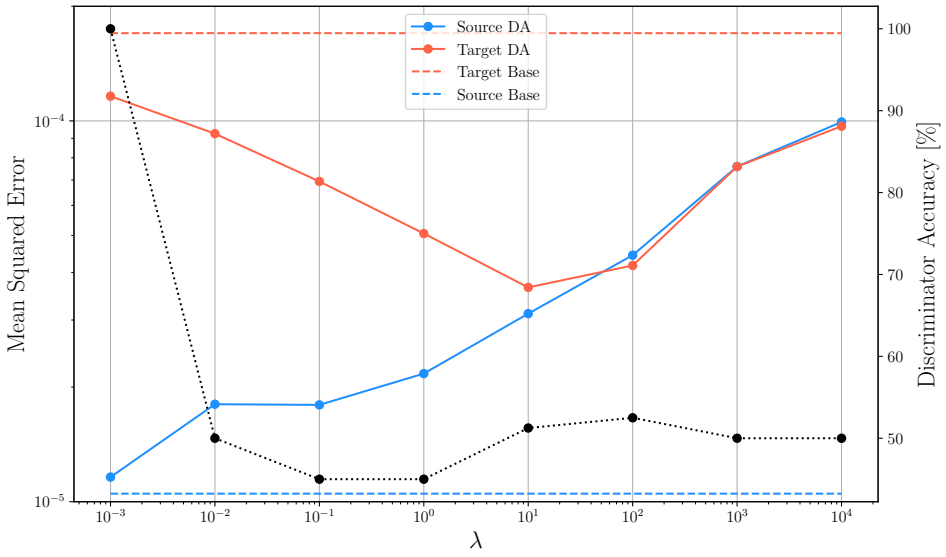
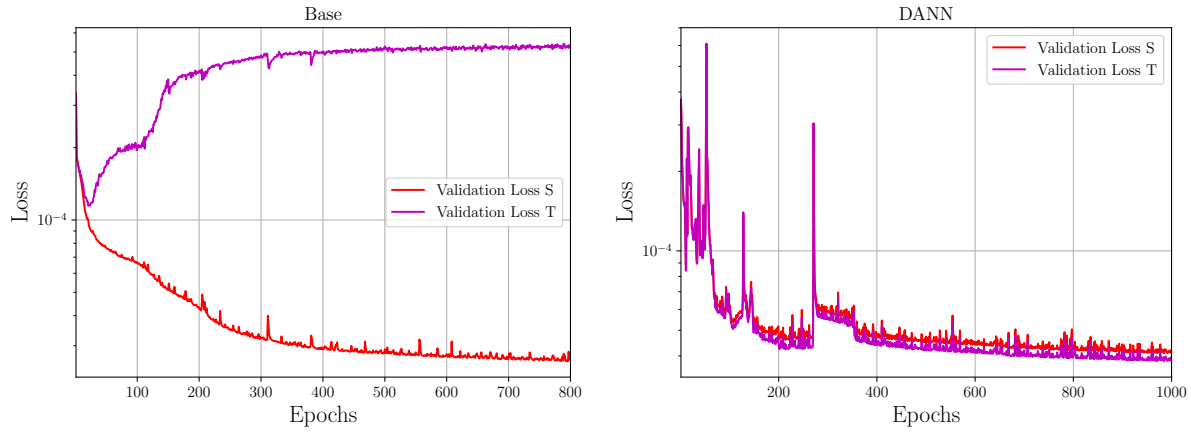


Figure 44: Influence of the divergence coefficient on the mean squared error. In black, the accuracy of the domain discriminator.

On figure 45, one can see how the loss evolves with and without domain adaptation. Without it, as soon as the model refines its reconstruction on source domain, the loss on target domain increases. Contrary to this, with domain adaptation, the two values decrease simultaneously.



(a) (b)

Figure 45: (a) Evolution of the test loss on both domains for the baseline experiment. (b) Evolution of the test loss on both domains for the domain adaptation experiment ($\lambda = 10^2$).

One can compare figure 43 and figure 46 to see the improvement brought by DA. Even if the reconstruction on target domain is not perfect and the one on source domain is slightly degraded, we see that the model has learned to inverse the target operator without labeled data. This is a clear demonstration of the power of IDR methods on inverse problems.

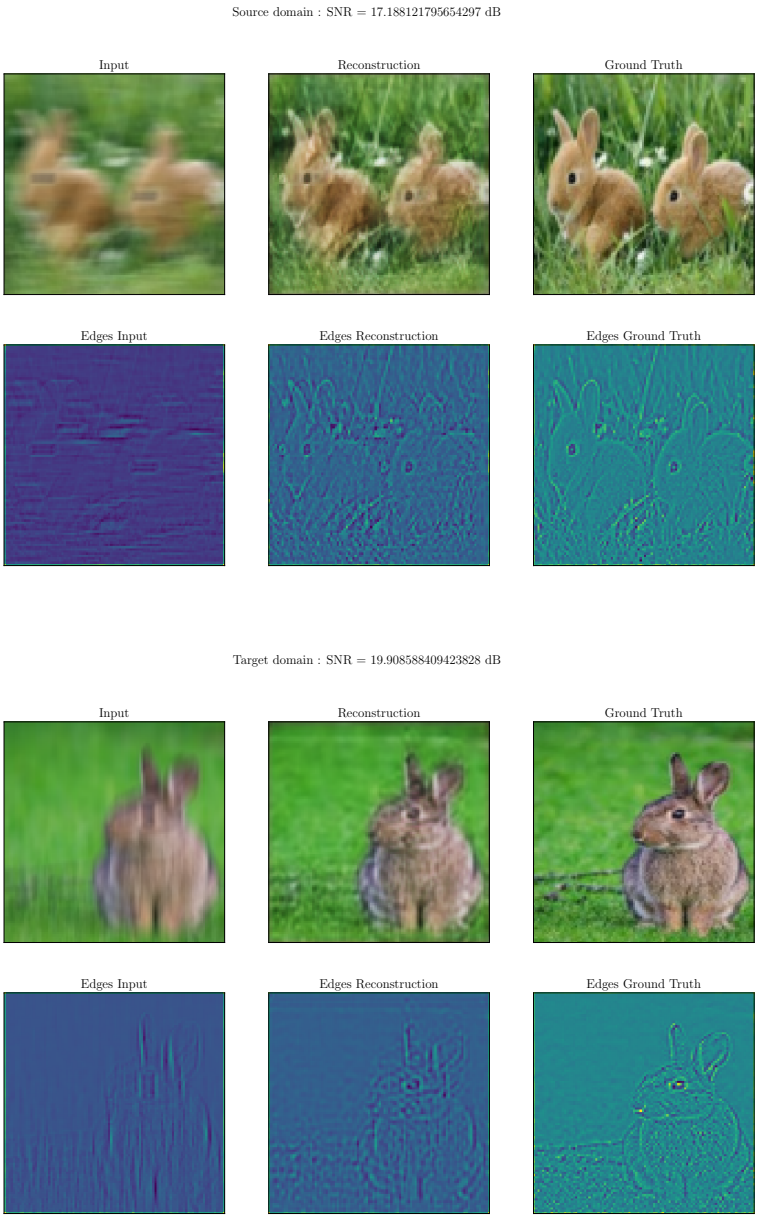


Figure 46: Domain adapatation results for the deblurring example. Reconstruction of a source image (top) and a target image (bottom) using the U-Net with a domain discriminator. ($\lambda = 10^1$)

When asking for too much invariance (left of the minimum on figure 44), the model loses its discriminative power. This can be seen on figure 47. The model is not able to correctly reconstruct the source or target images anymore, even if they seem to be invariant.

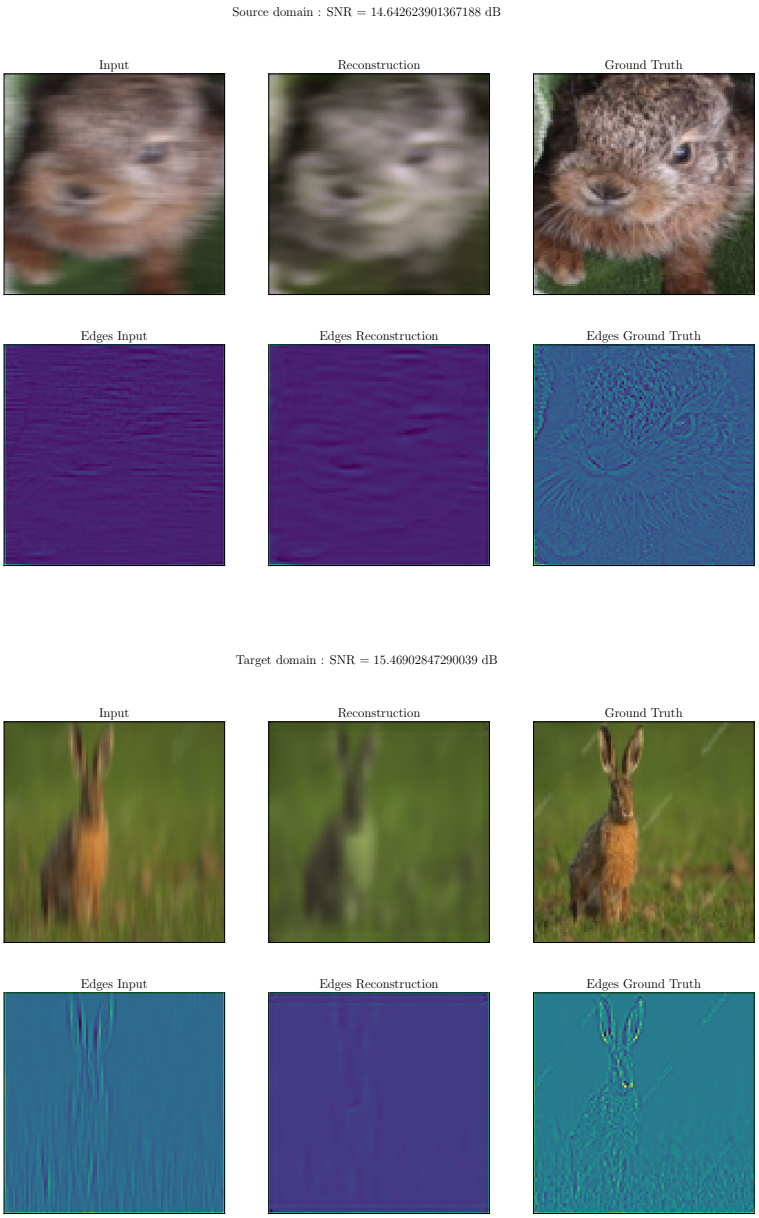


Figure 47: Domain adapatation results for the deblurring example. Reconstruction of a source image (top) and a target image (bottom) using the U-Net with a domain discriminator. ($\lambda = 10^3$)

From this example, we can conclude that IDR methods can be extended to the field of inverse problems. Here, a style shift is equivalent to a change of forward operator. The supervised part of the setting can be used to extract a prior on the instances. Then, the unsupervised part on target domain can be used to ensure data consistency on this domain.

6.5 Gravimetry and content shift

In the previous examples, we were always generating the shift by modifying the style component. In our theoretical analysis of IDR methods, we stated that a content shift cannot be reduced by IDR methods. We keep illustrating this with an inverse problem, gravimetry.

6.5.1 Dataset and shift

Without going into details, gravimetry is an inverse problem consisting of estimating a density distribution from a sparsely sampled gravitational field. For simplicity, we consider a two-dimensional density map made of randomly located shapes (with a resolution of 64x64). The gravitational field is obtained by convolving with the appropriate Green function. The result is subsampled to obtain a 16x16 grid.

The choice of shapes constituting the density field will generate our shift. Each map is made of either squares or circles. At the basis level (no shift), we consider two domains with the same proportion of squares and circles. To increase the shift, we progressively imbalanced the proportion to finally reach a complete shift, the source domain only contains circles and target domain only squares. The shift is described and shown on figure 48 and table 7.

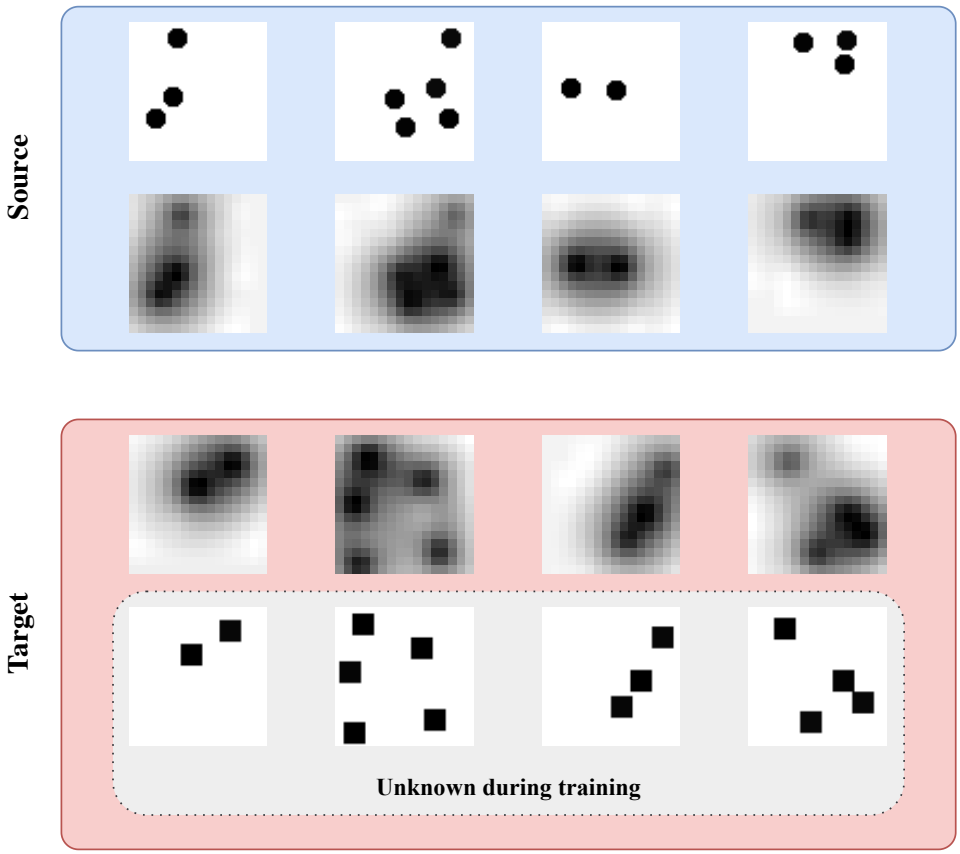


Figure 48: Preview of the considered dataset and shift for the gravimetry domain adaptation.

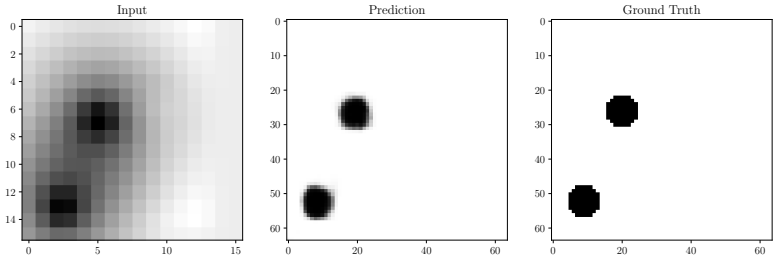
	Source domain		Target domain	
Shift	**Circle maps**	**Square maps**	**Circle maps**	**Square maps**
0	50 %	50 %	50 %	50 %
1	75 %	25 %	25 %	75 %
2	90 %	10 %	10 %	90 %
3	99 %	1 %	1 %	99 %
4	100 %	None	None	100 %

Table 7: Repartition of instances in the source and target domains for the gravimetry example.

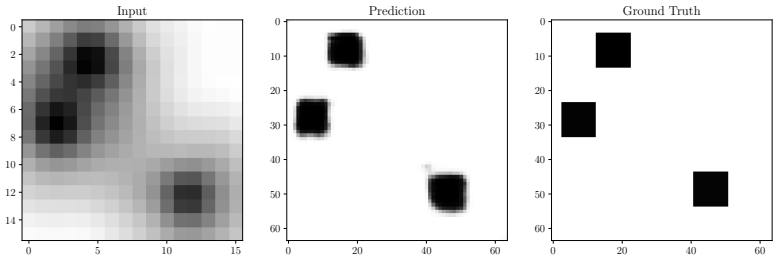
6.5.2 Results

As always, let us start by observing the need for domain adaptation to tackle the shift. If we regress the model only on source domain and then test it on target domain, we observe a clear drop in performance. This is shown on figure 50 where we consider the total shift (shift parameter equals 5). Additionally, we observe clearly that the model has not discovered the correct prior on the target domain by replacing the squares with circles. This is a clear demonstration of the content shift. For the situation without shift, the model is able to correctly reconstruct the map as shown on figure 49.

Here, we used a fully convolutional architecture for the encoder and the decoder. We restricted our study to the baseline and CORAL and DANN for domain adaptation. Other methods have been tested but have not shown any significant improvement.

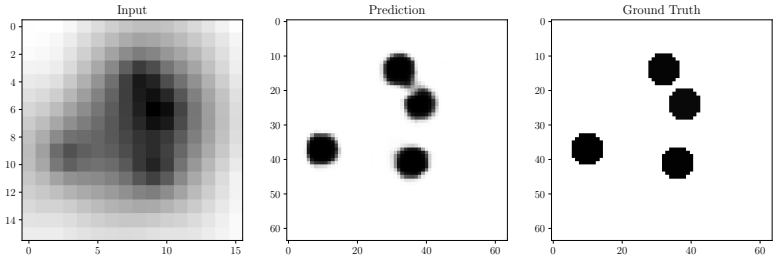


(a)

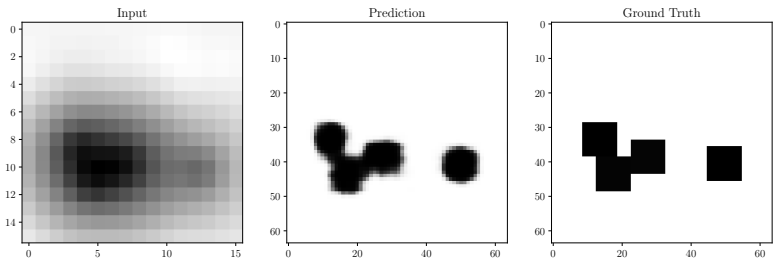


(b)

Figure 49: Reconstructions in the case without shift with the baseline architecture.



(a)

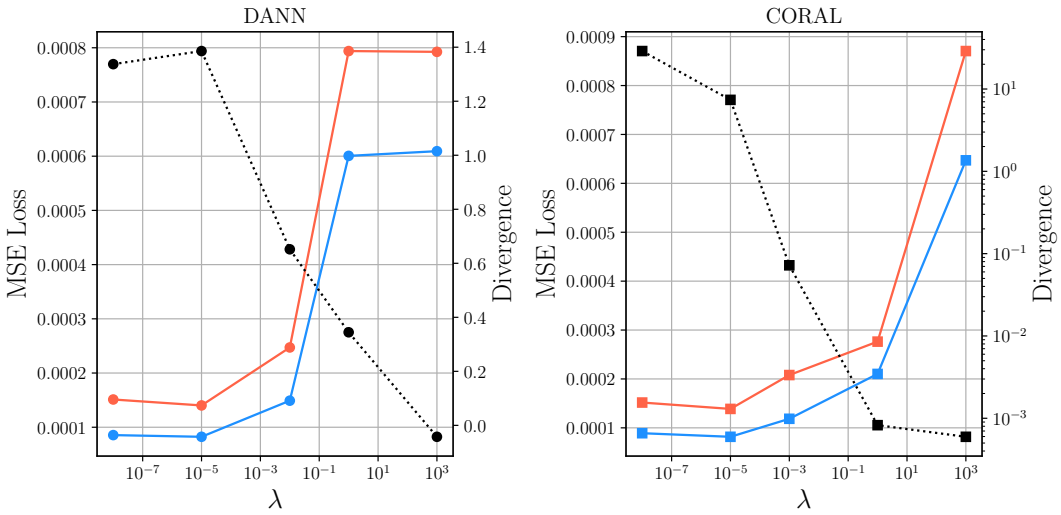


(b)

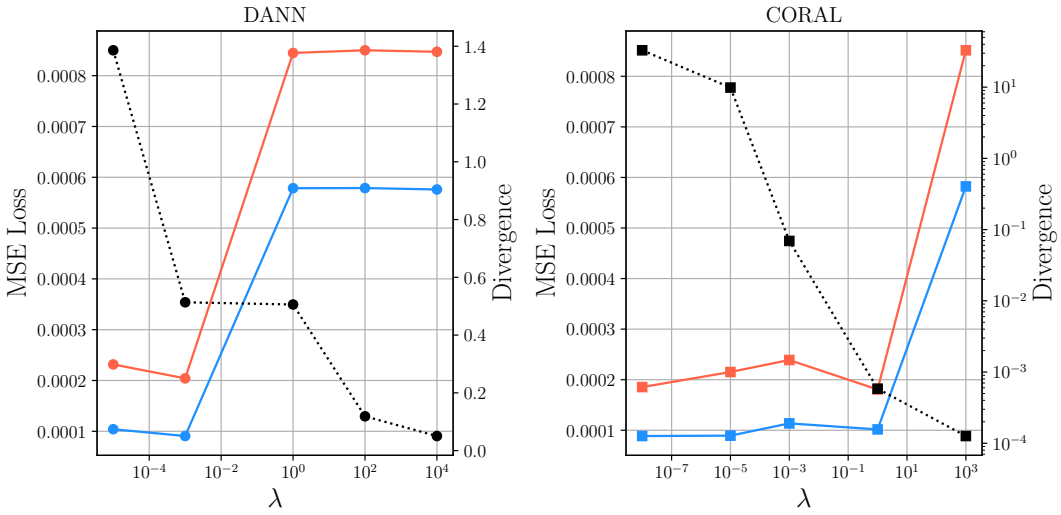
Figure 50: Reconstruction in the case of total shift with the DANN architecture.

On this example, as the shift is on the content level, the DA methods cannot help

reduce the drop in performance. This is shown on figure 52. The model is not able to correctly reconstruct the target domain and the loss does not go lower than the baseline approach.



(a)



(b)

Figure 51: Influence of the divergence coefficient on the mean squared error for the gravimetry example. In blue, MSE on source; in red, MSE on target domain; black dotted lines represent the evolution of the associated divergence.

This is clearer on figure 51. For a style shift, like the first example of digit classification, the performance on target domain can reach a maximum when tuning the trade-off between invariance and source domain accuracy (as on figure 30). Here, as we face content shift, we cannot find this trade-off. Either way, the divergence metric is high and the model can reconstruct the source but not the target. We are close to the baseline architecture. When the divergence starts decreasing, the loss on both domains increases without reaching a minimizer on target. When the divergence is low, the features extracted are

invariant but not useful regarding the task.

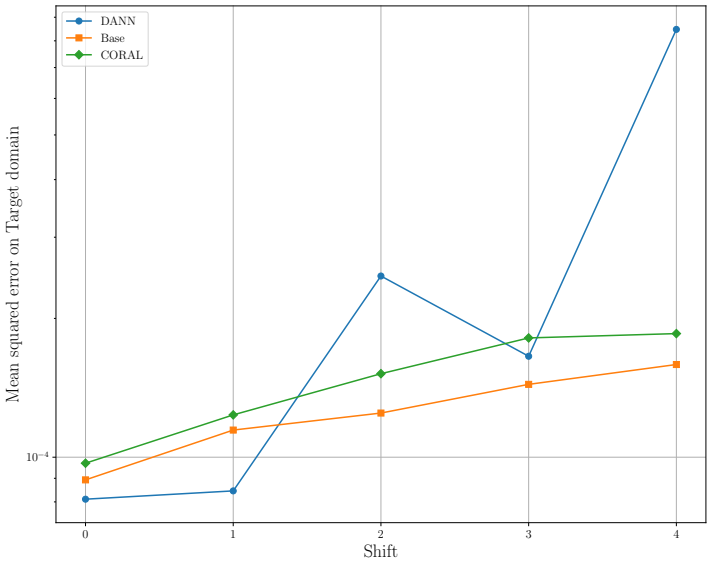


Figure 52: Mean squared error for the different methods depending on the shift for the gravimetry example.

In conclusion, we can see the impossibility of Domain adaptation to tackle a content shift. Even if this example is simplified and may seem obvious, it is important that one wonders about the nature of the shift when considering UDA. Here, we instantiate this for the IP framework but it can be extended to other fields. For the previous example, the source domain helped us deduce the prior of the content, which was common to both examples. Here, this is not possible as the prior itself is shifted.

7 Conclusion

This work has been a global study of distribution shifts and domain adaptation. By mixing theoretical aspects and numerical results, we have been able to provide an original view on DA and its associated methods. When dealing with distribution shift, the deep learning afficionado must be aware that he is somewhere between the ideal case (no shift) and pure infeasibility (impossible DA, for example).

On this path, we draw three important criteria for the shift to be tackled. The first, most obvious and least ambigous, is the performance on the source domain. If the model is not able to correctly classify the source domain, it will not be able to adapt to the target domain. The second one is intuitive as well; it states that either the initial inputs must share a similar distribution. If that is not the case, our model should be able to extract invariant features. This is not an easy task. On the one hand, it is not always possible. Indeed, we stated that this was feasible only when the shift was influencing irrelevant features of the covariate. We illustrated this later with the gravimetry example. On the other hand, asking a neural network to extract an invariant feature is not trivial. There are many divergences, metrics, and discrepancies that can be considered. All of them do not share the same properties, and computing them often requires additional tasks

(training a neural network, solving an optimization problem, *etc.*). Also, when requiring invariance too strongly, the model may lose its discriminative power. About this, we may have observed the fact that asking for invariance does not necessarily imply exact equality between distributions, but sufficient support overlap may be enough. The last criterion is the most subtle. It is the notion of a conditional shift. Even if we respect the covariate shift assumption and the covariates are not conditionally-shifted, by seeking invariance, we may induce a conditional shift. We illustrated this with an example. This is tightly related to the broad notion of out-of-distribution generalization and can be linked with the Information Bottleneck framework.

The framework of Unsupervised Domain Adaptation can be extended to the inverse problem framework. Shifting the noise or forward operator would lead to a style shift that can be solved by IDR methods. This is certainly a promising field that could be tested on more complex problems.

Regarding our numerical experiments, we showed the efficiency of several IDR methods on convenient shifts for image classification and inverse problem solving. We verified our analysis of impossible DA in two pathological cases: a content shift and a spuriously correlated example.

References

[Achille and Soatto, 2018] Achille, A. and Soatto, S. (2018). Emergence of Invariance and Disentanglement in Deep Representations. arXiv:1706.01350 [cs, stat].

[Ahuja et al., 2021] Ahuja, K., Caballero, E., Zhang, D., Gagnon-Audet, J.-C., Bengio, Y., Mitliagkas, I., and Rish, I. (2021). Invariance Principle Meets Information Bottleneck for Out-of-Distribution Generalization. In *Advances in Neural Information Processing Systems*, volume 34, pages 3438–3450. Curran Associates, Inc.

[Ando, 2004] Ando, R. K. (2004). Exploiting Unannotated Corpora for Tagging and Chunking. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 142–145, Barcelona, Spain. Association for Computational Linguistics.

[Arjovsky et al., 2017] Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein GAN. arXiv:1701.07875 [cs, stat].

[Bartlett et al., 2021] Bartlett, P. L., Montanari, A., and Rakhlin, A. (2021). Deep learning: a statistical viewpoint. arXiv:2103.09177 [cs, math, stat].

[Beery et al., 2018] Beery, S., van Horn, G., and Perona, P. (2018). Recognition in Terra Incognita.

[Ben-David et al., 2010] Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., and Vaughan, J. (2010). A theory of learning from different domains. *Machine Learning*, 79:151–175.

[Ben-David et al., 2006] Ben-David, S., Blitzer, J., Crammer, K., and Pereira, F. (2006). Analysis of Representations for Domain Adaptation. In *Advances in Neural Information Processing Systems*, volume 19. MIT Press.

[Bousmalis et al., 2016] Bousmalis, K., Trigeorgis, G., Silberman, N., Krishnan, D., and Erhan, D. (2016). Domain Separation Networks. arXiv:1608.06019 [cs].

[Carlucci et al., 2017] Carlucci, F. M., Porzi, L., Caputo, B., Ricci, E., and Bulò, S. R. (2017). AutoDIAL: Automatic DomaIn Alignment Layers. arXiv:1704.08082 [cs].

[Courty et al., 2017a] Courty, N., Flamary, R., Habrard, A., and Rakotomamonjy, A. (2017a). Joint Distribution Optimal Transportation for Domain Adaptation. arXiv:1705.08848 [cs, stat].

[Courty et al., 2017b] Courty, N., Flamary, R., Tuia, D., and Rakotomamonjy, A. (2017b). Optimal Transport for Domain Adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(9):1853–1865.

[Csurka, 2017] Csurka, G. (2017). Domain Adaptation for Visual Applications: A Comprehensive Survey. arXiv:1702.05374 [cs].

[Damodaran et al., 2018] Damodaran, B. B., Kellenberger, B., Flamary, R., Tuia, D., and Courty, N. (2018). DeepJDOT: Deep Joint Distribution Optimal Transport for Unsupervised Domain Adaptation. arXiv:1803.10081 [cs].

[David et al., 2010] David, S. B., Lu, T., Luu, T., and Pal, D. (2010). Impossibility Theorems for Domain Adaptation. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 129–136. JMLR Workshop and Conference Proceedings. ISSN: 1938-7228.

[Deng, 2012] Deng, L. (2012). The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]. *IEEE Signal Processing Magazine*, 29(6):141–142. Conference Name: IEEE Signal Processing Magazine.

[Durrieu et al., 2012] Durrieu, J.-L., Thiran, J.-P., and Kelly, F. (2012). Lower and upper bounds for approximation of the Kullback-Leibler divergence between Gaussian Mixture Models. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4833–4836, Kyoto, Japan. IEEE.

[Edwards, 2011] Edwards, D. A. (2011). On the Kantorovich–Rubinstein theorem. *Expositiones Mathematicae*, 29(4):387–398.

[Farahani et al., 2021] Farahani, A., Voghoei, S., Rasheed, K., and Arabnia, H. R. (2021). A Brief Review of Domain Adaptation. In Stahlbock, R., Weiss, G. M., Abou-Nasr, M., Yang, C.-Y., Arabnia, H. R., and Deligiannidis, L., editors, *Advances in Data Science and Information Engineering*, pages 877–894. Springer International Publishing, Cham.

[Fernando et al., 2013] Fernando, B., Habrard, A., Sebban, M., and Tuytelaars, T. (2013). Unsupervised Visual Domain Adaptation Using Subspace Alignment. In *2013 IEEE International Conference on Computer Vision*, pages 2960–2967. ISSN: 2380-7504.

[Flamary et al., 2021] Flamary, R., Courty, N., Gramfort, A., Alaya, M. Z., Boisbunon, A., Chambon, S., Chapel, L., Corenflos, A., Fatras, K., Fournier, N., Gautheron, L., Gayraud, N. T. H., Janati, H., Rakotomamonjy, A., Redko, I., Rolet, A., Schutz, A., Seguy, V., Sutherland, D. J., Tavenard, R., Tong, A., and Vayer, T. (2021). POT: Python Optimal Transport. *Journal of Machine Learning Research*, 22(78):1–8.

[Ganin and Lempitsky, 2015] Ganin, Y. and Lempitsky, V. (2015). Unsupervised Domain Adaptation by Backpropagation. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1180–1189. PMLR.

[Ganin et al., 2016] Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. (2016). Domain-Adversarial Training of Neural Networks. arXiv:1505.07818 [cs, stat].

[Gilton et al., 2021] Gilton, D., Ongie, G., and Willett, R. (2021). Model Adaptation for Inverse Problems in Imaging. arXiv:2012.00139 [cs, eess].

[Gong et al., 2012] Gong, B., Shi, Y., Sha, F., and Grauman, K. (2012). Geodesic flow kernel for unsupervised domain adaptation. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2066–2073. ISSN: 1063-6919.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. Adaptive computation and machine learning. The MIT Press, Cambridge, Massachusetts London, England.

[Goodfellow et al., 2014] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.

[Gopalan et al., 2011] Gopalan, R., Li, R., and Chellappa, R. (2011). Domain adaptation for object recognition: An unsupervised approach. In *2011 International Conference on Computer Vision*, pages 999–1006. ISSN: 2380-7504.

[Gretton et al., 2008] Gretton, A., Borgwardt, K., Rasch, M. J., Scholkopf, B., and Smola, A. J. (2008). A Kernel Method for the Two-Sample Problem. arXiv:0805.2368 [cs] version: 1.

[Gretton et al., 2012] Gretton, A., Sejdinovic, D., Strathmann, H., Balakrishnan, S., Pontil, M., Fukumizu, K., and Sriperumbudur, B. K. (2012). Optimal kernel choice for large-scale two-sample tests. In *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

[Guan and Liu, 2022] Guan, H. and Liu, M. (2022). Domain Adaptation for Medical Image Analysis: A Survey. *IEEE Transactions on Biomedical Engineering*, 69(3):1173–1185. Conference Name: IEEE Transactions on Biomedical Engineering.

[Gulrajani et al., 2017] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. (2017). Improved Training of Wasserstein GANs. arXiv:1704.00028 [cs, stat].

[Huang et al., 2006] Huang, J., Gretton, A., Borgwardt, K., Schölkopf, B., and Smola, A. (2006). Correcting Sample Selection Bias by Unlabeled Data. In *Advances in Neural Information Processing Systems*, volume 19. MIT Press.

[Japkowicz et al., 2000] Japkowicz, N., Hanson, S., and Gluck, M. (2000). Nonlinear Autoassociation Is Not Equivalent to PCA. *Neural Computation*, 12:531–545.

[Johansson et al., 2019] Johansson, F. D., Sontag, D., and Ranganath, R. (2019). Support and Invertibility in Domain-Invariant Representations. Publisher: [object Object] Version Number: 4.

[Kang et al., 2019] Kang, G., Jiang, L., Yang, Y., and Hauptmann, A. G. (2019). Contrastive Adaptation Network for Unsupervised Domain Adaptation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4888–4897, Long Beach, CA, USA. IEEE.

[Kawaguchi et al., 2023] Kawaguchi, K., Deng, Z., Ji, X., and Huang, J. (2023). How Does Information Bottleneck Help Deep Learning? arXiv:2305.18887 [cs, math].

[Kingma and Welling, 2022] Kingma, D. P. and Welling, M. (2022). Auto-Encoding Variational Bayes. arXiv:1312.6114 [cs, stat].

[Kirsch et al., 2021] Kirsch, A., Lyle, C., and Gal, Y. (2021). Unpacking Information Bottlenecks: Unifying Information-Theoretic Objectives in Deep Learning. arXiv:2003.12537 [cs, stat].

[Kong et al., 2023] Kong, L., Xie, S., Yao, W., Zheng, Y., Chen, G., Stojanov, P., Akinwande, V., and Zhang, K. (2023). Partial Identifiability for Domain Adaptation. arXiv:2306.06510 [cs, stat].

[Kouw and Loog, 2019] Kouw, W. M. and Loog, M. (2019). An introduction to domain adaptation and transfer learning. arXiv:1812.11806 [cs, stat].

[Kouw and Loog, 2021] Kouw, W. M. and Loog, M. (2021). A Review of Domain Adaptation without Target Labels. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(3):766–785.

[Lemberger and Panico, 2020] Lemberger, P. and Panico, I. (2020). A Primer on Domain Adaptation. arXiv:2001.09994 [cs, stat].

[Liu et al., 2023] Liu, H., Li, B., Lu, M., and Wu, Y. (2023). Real-World Image Deblurring via Unsupervised Domain Adaptation. pages 148–159.

[Liu et al., 2022] Liu, X., Yoo, C., Xing, F., Oh, H., Fakhri, G. E., Kang, J.-W., and Woo, J. (2022). Deep Unsupervised Domain Adaptation: A Review of Recent Advances and Perspectives. arXiv:2208.07422 [cs, eess].

[Long et al., 2015] Long, M., Cao, Y., Wang, J., and Jordan, M. I. (2015). Learning Transferable Features with Deep Adaptation Networks. arXiv:1502.02791 [cs] version: 2.

[Mansour et al., 2009] Mansour, Y., Mohri, M., and Rostamizadeh, A. (2009). Domain Adaptation: Learning Bounds and Algorithms. arXiv:0902.3430 [cs].

[Müller, 1997] Müller, A. (1997). Integral Probability Metrics and Their Generating Classes of Functions. *Advances in Applied Probability*, 29(2):429–443. Publisher: Applied Probability Trust.

[Nguyen et al., 2021a] Nguyen, A., Tran, T., Gal, Y., Torr, P. H. S., and Baydin, A. G. (2021a). KL Guided Domain Adaptation. *ArXiv*.

[Nguyen et al., 2021b] Nguyen, A. T., Tran, T., Gal, Y., and Baydin, A. G. (2021b). Domain Invariant Representation Learning with Domain Density Transformations. In *Advances in Neural Information Processing Systems*, volume 34, pages 5264–5275. Curran Associates, Inc.

[Pan and Yang, 2010] Pan, S. J. and Yang, Q. (2010). A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359.

[Ramponi and Plank, 2020] Ramponi, A. and Plank, B. (2020). Neural Unsupervised Domain Adaptation in NLP—A Survey. arXiv:2006.00632 [cs].

[Redko et al., 2019] Redko, I., Morvant, E., Habrard, A., Sebban, M., and Bennani, Y. (2019). *Advances in domain adaptation theory*. ISTE Press Ltd ; Elsevier Ltd, London, UK, Kidlington, oxford, UK. OCLC: 1131721824.

[Roark and Bacchiani, 2003] Roark, B. and Bacchiani, M. (2003). Supervised and unsupervised PCFG adaptation to novel domains. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 205–212.

[Ronneberger et al., 2015] Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation.

[Rosenfeld et al., 2021] Rosenfeld, E., Ravikumar, P., and Risteski, A. (2021). The Risks of Invariant Risk Minimization. arXiv:2010.05761 [cs, stat].

[Saenko et al., 2010] Saenko, K., Kulis, B., Fritz, M., and Darrell, T. (2010). Adapting Visual Category Models to New Domains. In Daniilidis, K., Maragos, P., and Paragios, N., editors, *Computer Vision – ECCV 2010*, Lecture Notes in Computer Science, pages 213–226, Berlin, Heidelberg. Springer.

[Sagawa et al., 2022] Sagawa, S., Koh, P. W., Lee, T., Gao, I., Xie, S. M., Shen, K., Kumar, A., Hu, W., Yasunaga, M., Marklund, H., Beery, S., David, E., Stavness, I., Guo, W., Leskovec, J., Saenko, K., Hashimoto, T., Levine, S., Finn, C., and Liang, P. (2022). Extending the WILDS Benchmark for Unsupervised Adaptation. arXiv:2112.05090 [cs, stat].

[Shen et al., 2018] Shen, J., Qu, Y., Zhang, W., and Yu, Y. (2018). Wasserstein Distance Guided Representation Learning for Domain Adaptation. arXiv:1707.01217 [cs, stat].

[Shimodaira, 2000] Shimodaira, H. (2000). Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of Statistical Planning and Inference*, 90(2):227–244.

[Shu et al., 2018] Shu, R., Bui, H. H., Narui, H., and Ermon, S. (2018). A DIRT-T Approach to Unsupervised Domain Adaptation.

[Shwartz-Ziv and Tishby, 2017] Shwartz-Ziv, R. and Tishby, N. (2017). Opening the Black Box of Deep Neural Networks via Information. arXiv:1703.00810 [cs].

[Singhal and Majumdar, 2020] Singhal, V. and Majumdar, A. (2020). A domain adaptation approach to solve inverse problems in imaging via coupled deep dictionary learning. *Pattern Recognition*, 100:107163.

[Sugiyama et al., 2007] Sugiyama, M., Nakajima, S., Kashima, H., Buenau, P., and Kawanabe, M. (2007). Direct Importance Estimation with Model Selection and Its Application to Covariate Shift Adaptation. In *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc.

[Sun et al., 2016] Sun, B., Feng, J., and Saenko, K. (2016). Correlation Alignment for Unsupervised Domain Adaptation.

[Sun and Saenko, 2016] Sun, B. and Saenko, K. (2016). Deep CORAL: Correlation Alignment for Deep Domain Adaptation. In Hua, G. and Jégou, H., editors, *Computer Vision – ECCV 2016 Workshops*, Lecture Notes in Computer Science, pages 443–450, Cham. Springer International Publishing.

[Tishby et al., 2000] Tishby, N., Pereira, F. C., and Bialek, W. (2000). The information bottleneck method. arXiv:physics/0004057.

[Tishby and Zaslavsky, 2015] Tishby, N. and Zaslavsky, N. (2015). Deep Learning and the Information Bottleneck Principle. arXiv:1503.02406 [cs].

[Tzeng et al., 2014] Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., and Darrell, T. (2014). Deep Domain Confusion: Maximizing for Domain Invariance. arXiv:1412.3474 [cs].

[Vapnik, 2000] Vapnik, V. N. (2000). *The Nature of Statistical Learning Theory*. Springer New York, New York, NY.

[Wahba, 2003] Wahba, G. (2003). An introduction to reproducing kernel hilbert spaces and why they are so useful. *IFAC Proceedings Volumes*, 36(16):525–528.

[Wang and Deng, 2018] Wang, M. and Deng, W. (2018). Deep Visual Domain Adaptation: A Survey. *Neurocomputing*, 312:135–153. arXiv:1802.03601 [cs].

[Wang et al., 2024] Wang, Y., Hazimeh, H., Ponomareva, N., Kurakin, A., Hammoud, I., and Arora, R. (2024). DART: A Principled Approach to Adversarially Robust Unsupervised Domain Adaptation. arXiv:2402.11120 [cs, stat] version: 1.

[Wilson and Cook, 2020] Wilson, G. and Cook, D. J. (2020). A Survey of Unsupervised Deep Domain Adaptation. *ACM Transactions on Intelligent Systems and Technology*, 11(5):51:1–51:46.

[Xu and Raginsky, 2017] Xu, A. and Raginsky, M. (2017). Information-theoretic analysis of generalization capability of learning algorithms. arXiv:1705.07809 [cs, math, stat].

[Yu et al., 2023] Yu, X., Tseng, H.-H., Yoo, S., Ling, H., and Lin, Y. (2023). INSURE: An Information Theory Inspired Disentanglement and Purification Model for Domain Generalization. arXiv:2309.04063 [cs].

[Yue et al., 2023] Yue, Z., Sun, Q., and Zhang, H. (2023). Make the U in UDA Matter: Invariant Consistency Learning for Unsupervised Domain Adaptation. *Advances in Neural Information Processing Systems*, 36:26991–27004.

[Zhao et al., 2019] Zhao, H., Combes, R. T. D., Zhang, K., and Gordon, G. (2019). On Learning Invariant Representations for Domain Adaptation. In *Proceedings of the 36th International Conference on Machine Learning*, pages 7523–7532. PMLR. ISSN: 2640-3498.

A Ommitted methods

- **Deep Separation Network** [Bousmalis et al., 2016] : This method is very similar to DDC (as it is based on a MMD minimization) but it adds a reconstruction network aiming at reconstruct the initial data point. In practice, it has been implemented and compared with our other methods but it did not show any significant improvement compared to DDC. DSN can be seen as an IDR method but it overcomplicates the architecture and does not provide any additional insight. Regarding the interpretatbility of the latent space, the reconstruction part resembles the complement encoder part introduced when discussing spurious correlations.

- **Invariant CONsistency learning** [Yue et al., 2023] : This very recent and award-winning ([Sagawa et al., 2022]) method seems to be very promising and take a step outward the IDR approach. However, it is restricted to classification tasks and, therefore, does not fit our versatility requirement. Without further explanation, the idea is to apply clustering on the target domain before assigning a class to the cluster.

- **Information iNspired diSentanglement and pURification modEl (INSURE)** [Yu et al., 2023] : This method may also fit also the IDR structure even if it has been created for domain generalization and not domain adaptation. However, we think that it may look like the DANN approach. The additional purification step echoes the notion of redundant information that will be of interest in the information bottleneck approach of UDA.

- **Virtual Adversarial Domain Adaptation (VADA)** [Shu et al., 2018] : Similarly, this method is a direct extension of DANN as it used the same adversarial training procedure. They add a clustering aspect for source domain to improve disposition of the latent space. Consequently, it is not applicable to regression tasks. In the same idea, the method presented in [Kang et al., 2019] promotes clustering.

- **AUTOmatic Domain alIgnment Layers (AUTODIAL)** [Carlucci et al., 2017] : AUTODIAL may fit the IDR framework also. Yet, it is not based on computing a divergence and directly minimizing it by adding it to the loss term. Instead, the alignment is forced by specific batch normalization layers. Without going into details of the method, the statistics of the normalization layers are separated between contaminated source and target domains. A mixture parameter is learned to control the alignment.

- **Transformation based methods** : [Gong et al., 2012, Fernando et al., 2013] : We consider that estimating transformation operator between domain is similar to the optimal transport idea. Then, DeepJDOT provides a proper instance of that class of methods.

B Notations

- $\mathcal{S}, \mathcal{T}$: Source and target joints distributions

- $\mathcal{S}_X, \mathcal{T}_X$: Marginal distributions of the inputs
- $\mathcal{S}_{Y|X}, \mathcal{T}_{Y|X}$: Conditional distributions of the outputs given the inputs
- $\mathcal{S}_Y, \mathcal{T}_Y$: Marginal distributions of the outputs
- $\hat{\mathcal{S}}, \hat{\mathcal{T}}$: Empirical distributions
- S, T : Densities of the joint distributions
- S_X, T_X : Densities of the marginal distributions of the inputs
- $S_{Y|X}, T_{Y|X}$: Densities of the conditional distributions of the outputs given the inputs
- $\mathfrak{S}$, $\mathfrak{T}$: Source and target datasets
- $W(\cdot, \cdot)$: Wasserstein distance
- $KL(\cdot||\cdot)$: Kullback-Leibler divergence
- $TV(\cdot, \cdot)$: Total Variation distance
- $JS(\cdot, \cdot)$: Jensen-Shannon divergence

C Proofs

C.1 Proof of lemma 2.5

Proof. From the Kantorovich-Rubinstein duality (with $K = 1$), we have the following if $\mathcal{S}_{Y|X} = \mathcal{T}_{Y|X}$.

$$\begin{aligned}
W(\mathcal{S}, \mathcal{T}) &= \sup_{||f||_{\mathscr{L}} \leq 1} \left| \int_X \int_Y f(x,y)(S(x,y) - T(x,y))dxdy \right| \\
&= \sup_{||f||_{\mathscr{L}} \leq 1} \left| \int_X \int_Y f(x,y)(S_{Y|X}(x,y)S_X(x) - T_{Y|X}(x,y)T_X(x))dxdy \right| \\
&= \sup_{||f||_{\mathscr{L}} \leq 1} \left| \int_X \int_Y f(x,y)S_{Y|X}(x,y)(S_X(x) - T_X(x))dxdy \right| \\
&= \sup_{||f||_{\mathscr{L}} \leq 1} \left| \int_X \underbrace{\mathbb{E}_{Y|X}[f(x,y)]}_{g(x)}(S_X(x) - T_X(x))dx \right|
\end{aligned}$$

Now, we must prove that under the smoothness assumption of the conditional distribution, g is Lipschitz-continuous.

$$
\begin{aligned}
|g(x) - g(x')| &= \left|\mathbb{E}_{Y|X=x}[f(x,y)] - \mathbb{E}_{Y|X=x'}[f(x',y)]\right| \\
&= \left|\int_Y f(x,y)S_{Y|X}(x,y) - f(x',y)S_{Y|X}(x',y))dy\right| \\
&= \left|\int_Y (f(x,y) - f(x',y))S_{Y|X}(x,y) + f(x',y)S_{Y|X}(x,y) - f(x',y)S_{Y|X}(x',y)dy\right| \\
&\leq \int_Y |f(x,y) - f(x',y)|S_{Y|X}(x,y)dy + \left|\int_Y f(x',y)(S_{Y|X}(x,y) - S_{Y|X}(x',y))dy\right| \\
&\leq d(x,x') + W(\mathcal{S}_{Y|X=x}, \mathcal{S}_{Y|X=x'}) \leq (1+K)d(x,x')
\end{aligned}
$$

So, we have that g is $(1+K)$-Lipschitz continuous. By introducing it in our previous equation, we have the following.

$$
W(\mathcal{S},\mathcal{T}) \leq \sup_{||g||_{\mathscr{L}}\leq 1+K} \left|\int_X g(x)(S_X(x) - T_X(x))dx\right| = (1+K)W(\mathcal{S}_X, \mathcal{T}_X)
$$

□

C.2 Proof of lemma 2.6

Proof. From the Kantorovich-Rubinstein duality (with $K = 1$), we have the following.

$$
\begin{aligned}
W(\mathcal{S},\mathcal{T}) &= \sup_{||f||_{\mathscr{L}}\leq 1} \left|\int_X \int_Y f(x,y)(S(x,y) - T(x,y))dxdy\right| \\
&= \sup_{||f||_{\mathscr{L}}\leq 1} \left|\int_X \int_Y f(x,y)(S_{Y|X}(x,y)S_X(x) - T_{Y|X}(x,y)T_X(x))dxdy\right| \\
&= \sup_{||f||_{\mathscr{L}}\leq 1} \left|\int_X \left(\int_Y f(x,y)S_{Y|X}(x,y)S_X(x)dy\right) - \left(\int_Y f(x,y)T_{Y|X}(x,y)T_X(x)dy\right) dx\right| \\
&= \sup_{||f||_{\mathscr{L}}\leq 1} \left|\int_X \mathbb{E}_{S_{Y|X}}[f(x,y)]S_X(x) - \mathbb{E}_{T_{Y|X}}[f(x,y)]T_X(x)dx\right|
\end{aligned}
$$

We can use the KR duality again on the conditional distributions :

$$
\forall x \in X, \left|\mathbb{E}_{S_{Y|X}}[f(x,y)] - \mathbb{E}_{T_{Y|X}}[f(x,y)]\right| \leq W(\mathcal{S}_{Y|X}, \mathcal{T}_{Y|X})
$$

After that, we can arbitrarily replace the expectation along $\mathcal{S}$ or $\mathcal{T}$. For example, here, we choose to expand it along $\mathcal{T}$.

$$
\begin{aligned}
W(\mathcal{S},\mathcal{T}) &\leq \sup_{||f||_{\mathscr{L}}\leq 1} \left| \int_X \underbrace{\mathbb{E}_{S_{Y|X}}[f(x,y)]}_{g(x)} S_X(x) - \mathbb{E}_{S_{Y|X}}[f(x,y)]T_X(x) + W(\mathcal{S}_{Y|X},\mathcal{T}_{Y|X})T_X(x)dx \right| \\
&= \sup_{||g||_{\mathscr{L}}\leq 1+K} \left| \int_X g(x)(S_X(x) - T_X(x)) + W(\mathcal{S}_{Y|X},\mathcal{T}_{Y|X})T_X(x)dx \right| \\
&\leq (1+K)W(\mathcal{S}_X,\mathcal{T}_X) + \mathbb{E}_{T_X}[W(\mathcal{S}_{Y|X},\mathcal{T}_{Y|X})]
\end{aligned}
$$

As the second term with the expectation was arbitrarily chosen to be with respect to the target distribution, we finally deduce the following.

$$
W(\mathcal{S},\mathcal{T}) \leq (1+K)W(\mathcal{S}_X,\mathcal{T}_X) + \min\Big\{\mathbb{E}_{S_X}[W(\mathcal{S}_{Y|X},\mathcal{T}_{Y|X}), \mathbb{E}_{T_X}[W(\mathcal{S}_{Y|X}],\mathcal{T}_{Y|X})]\Big\}
$$

□

C.3 Proof of theorem 3.4

Proof. By using the Kantorovich-Rubinstein duality, we have the following.

$$
\begin{aligned}
|R_S(h) - R_T(h)| = |\mathbb{E}_{\mathcal{S}}[\ell_h x, y)] - \mathbb{E}_{\mathcal{T}}[\ell_h(x,y)]| &\leq ||\ell_h||_{\mathscr{L}} W(\mathcal{S},\mathcal{T}) \\
\implies R_S(h) &\leq R_T(h) + ||\ell_h||_{\mathscr{L}} W(\mathcal{S},\mathcal{T})
\end{aligned}
$$

By lemma 2.5, we can expand the distance between joint distributions.

$$
R_S(h) \leq R_T(h) + ||\ell_h||_{\mathscr{L}}(1+K)\Big(W(\mathcal{S}_X,\mathcal{T}_X) + \min\Big\{\mathbb{E}_{S_X}[W(\mathcal{S}_{Y|X},\mathcal{T}_{Y|X}), \mathbb{E}_{T_X}[W(\mathcal{S}_{Y|X}],\mathcal{T}_{Y|X})]\Big\}\Big)
$$

We deduce now that the constant of interest is linked with the Lipschitz constant of ℓ_h. Yet, we can expand its expression to obtain a more convenient form involving only the Lipschitz constant of the model itself.

$$
\begin{aligned}
d(\ell_h(x_1,y_1);\ell_h(x_2,y_2)) &\leq ||\ell||_{\mathscr{L}}\big(d(h(x_1),y_1;h(x_2),y_2)\big) \\
&\leq ||\ell||_{\mathscr{L}}\big(d(h(x_1);h(x_2)) + d(y_1,y_2)\big) \text{ by triangle inequality} \\
&\leq ||\ell||_{\mathscr{L}}\big(||h||_{\mathscr{L}} d(x_1;x_2) + d(y_1,y_2)\big) \\
&\leq ||\ell||_{\mathscr{L}}(||h||_{\mathscr{L}} + 1)d(x_1,y_1;x_2,y_2)
\end{aligned}
$$

From this, we can upper bound the constant C by $||\ell||_{\mathscr{L}}(||h||_{\mathscr{L}} + 1)(1+K)$. Finally, we obtain the following.

$$
\begin{aligned}
R_S(h) \leq & R_T(h) \\
&+ \underbrace{||\ell||_{\mathscr{L}}(||h||_{\mathscr{L}} + 1)(1+K)}_{C}\Big(W(\mathcal{S}_X,\mathcal{T}_X) + \min\Big\{\mathbb{E}_{S_X}[W(\mathcal{S}_{Y|X},\mathcal{T}_{Y|X}), \mathbb{E}_{T_X}[W(\mathcal{S}_{Y|X}],\mathcal{T}_{Y|X})]\Big\}\Big)
\end{aligned}
$$

□

C.4 Proof of proposition 5.1

Proof. From its definition, one can rewrite conditional mutual information as follows.

$$I(X;Y|Z) = H(X|Z) - H(X|Y,Z)$$

As entropy is always positive, we have the following.

$$I(X;Y|Z) \leq H(X|Z)$$

However, if the encoder is invertible, there exists a deterministic decoder that can recover the input from the latent representation. This means that $H(X|Z) = 0$ and therefore $I(X;Y|Z) = 0$. □

C.5 Proof of theorem 5.1

Proof. Firstly, we recall one of the definitions of mutual information. The latter can be expressed as the Kullback-Leibler divergence between the joint distribution and the product of the marginal distributions (hypothetical joint distribution if random variables are independent). Without loss of generality, we consider the source distributions. This yields the following.

$$\begin{aligned} I(X;Y|Z) = D_{KL}(\mathcal{S}_{XY|Z}||\mathcal{S}_{X|Z}\mathcal{S}_{Y|Z}) = 0 \\ \implies \mathcal{S}_{XY|Z} = \mathcal{S}_{X|Z}\mathcal{S}_{Y|Z} \\ \implies \frac{\mathcal{S}_{XY|Z}}{\mathcal{S}_{X|Z}} = \mathcal{S}_{Y|Z} \\ \implies \mathcal{S}_{Y|Z} = \mathcal{S}_{Y|Z,X} \end{aligned}$$

In other words, once conditioned to Z the label Y is independent of the input X. If we express the conditional distribution as follows (same expression as in [Johansson et al., 2019]).

$$S(y|z) = \frac{\int_{x \in g^{-1}(z)} S(y|x)S(x)dx}{\int_{x \in g^{-1}(z)} S(x)dx}$$

Considering the fact that $S(y|x) = S(y|x,z)$, we can chose an arbitrary $x' \in g^{-1}(z)$, and we have that $S(y|x) = S(y|x'), \forall x \in g^{-1}(z)$. Then, the conditional term leave the integral. And finally injecting the covariate shift assumption, we obtain the following being valid for all z.

$$S(y|z) = S(y|x') = T(y|x') = T(y|z)$$

□

D Implementation details

We will now detail the choices that have been made for the implementation of the different methods. The code has been written in Python with the help of the following packages.

- Deep learning framework : PyTorch
- Data manipulation : Numpy
- Optimal transport solver : POT (Python Optimal Transport) [Flamary et al., 2021]
- Convex optimization for DAN : CVXPY

D.1 Noisy digit classification

Dataset The dataset is composed of the MNIST dataset [Deng, 2012] following its train/test split. Namely, the train set is constituted of 60000 samples and 2000 for the test.

Architecture The common architecture for all methods (encoder and head) is a convolutional neural network described in table 8. The batch size has been chosen to be 2048.

Encoder

Convolutional	**Channels**	**Kernel size**
Conv1	10	5
Conv2	20	5
Conv3	20	5
Dense	**In features**	**Out features**
Flat & Drop	Probability 0.5	
Dense1	9680	50
Dense2	50	15
Batch normalization		

Head

Dense	**In features**	**Out features**
Dense1	15	30
Dense2	30	10

Table 8: Chosen architecture for the noisy digit classification.

Regarding the additional neural networks, the critic for WDGRL and the domain discriminator in DANN, they are both made dense layers.

Additional Network		
Dense	**In features**	**Out features**
Dense1	15	30
Dense2	30	30
Dense3	30	1

Table 9: Chosen architecture for the additional networks in the noisy digit classification.

Training procedure For all the methods, the chosen optimizer is Adam. The learning rates and the number of epochs have been tuned to obtain the best results from each method.

Method specific aspects Here are some specific details for methods that require additional implementation.

- DDC : We used four radial basis functions with bandwidths of 0.01, 0.1, 1.0 and 10.0.
- DAN : The radial basis function are similar to DDC.
- WDGRL : The critic is trained for 5 epochs after each epoch of the encoder. The learning rate of the critic is 0.01.
- DeepDJOT : The discrete distributions was chosen to be uniform over the support of the data. Consequently, the optimal transport problem boils down to an assignment problem.

D.2 Spuriously correlated digit classification

Architecture The architecture is a simplification of the previous one (as it consists now in binary classification).

Encoder

Convolutional	**Channels**	**Kernel size**
Conv1	10	5
Conv2	20	5
Dense	**In features**	**Out features**
Flat & Drop	Probability 0.5	
Dense1	11520	50
Dense2	50	10
Batch normalization		

Head

Dense	**In features**	**Out features**
Dense1	10	30
Dense2	30	10

Table 10: Chosen architecture for the spurious correlated digit classification.

For the additional architecture for complementary encoding, the second encoder has the same structure and the decoder has the follwoing architecture.

Decoder

Dense	**In features**	**Out features**
Dense1	10	50
Dense2	50	1000
Convolutional	**Channels**	**Kernel size**
ConvT1	10	20
ConvT2	20	1

Table 11: Chosen architecture for the decoder in the spurious correlated digit classification.

Mutual information Mutual information has been computed with the scikit-learn package. For the residual information, the conditional aspect have been done by binning the latent representation. Then, we take the average over the bins.

D.3 Deblurring and operator shift

Architecture We used a U-Net architecture made of three levels of convolutional layers.

Encoder 1		
Convolutional	**Channels**	**Kernel size**
Conv1	16	3
Conv2	16	3
Max Pool	16	2
Encoder 2		
Conv1	16	3
Conv2	16	3
Max Pool	16	2
Bottleneck		
Conv1	16	3
Conv2	16	3
Upsample	16	2
Decoder 2		
Conv1	32	3
Conv2	16	3
Upsample	16	2
Decoder 1		
Conv1	32	3
Conv2	16	3
Upsample	16	2

Table 12: Chosen architecture for the deblurring example

For the discriminator, the architecture is convolutional and described on table 13.

Discriminator		
Convolutional	**Channels**	**Kernel size**
Conv1	16	3
Max pool	16	2
Conv2	16	3
Max pool	16	2
Conv3	16	3
Dense	**In features**	**Out features**
Flat & Drop	Probability 0.5	
Dense1	10000	200
Dense2	200	10
Dense3	10	1

Table 13: Chosen architecture for the discriminator in the deblurring example.

The reconstruction loss is the mean squared error between the input and the output of the U-Net plus an edge fidelity term. The latter is computed with a 3x3 laplacian of Gaussian filter. The batch size is 64.

Dataset The dataset is made of 2600 training and validation images and 200 test samples of size 128x128. They have been gathered on the several free datasets available on the internet, https://images.cv/dataset/rabbit-image-classification-dataset, https://www.kaggle.com/datasets/muniryadi/cat-vs-rabbit. The motion blur is generated by a 11x11 pixels kernel.

D.4 Gravimetry and content shift

Architecture The architecture is a fully convolutional network described on table 14. The discriminator the same as the one used on the deblurring example (table 13).

Encoder

Convolutional	**Channels**	**Kernel size**
Conv1	8	3
Conv2	16	3
Conv3	16	3

Head

Convolutional	**Channels**	**Kernel size**
Conv1	16	3
Conv2	8	3
Conv3	1	3

Table 14: Chosen architecture for the gravimetry example.

Dataset The dataset is made of 20000 training and validation examples and 500 test samples of size 64x64. The gravimetric data are generated by convolution with a 6x6 kernel.

D.5 Computing the Lipschitz constant of a neural network

To obtain an approximation of the Lipschitz constant of neural network, we used the idea [Arjovsky et al., 2017]. We compute the gradient of the network at different points (on the dataset and on radom convex combinations of the data points from source and target). The Lipschitz constant should be the supremum norm of the gradient. We estimate it by taking the L2 norm of the evaluated gradients.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**