Deep diversity learning for better generalization to unseen domains

Thomas Duboudin

ÉCOLE CENTRALE LYON LIRIS THALES

THÈSE DE DOCTORAT DE L'UNIVERSITÉ DE LYON
opérée au sein de l'École Centrale de Lyon

Délivrée par l'École Doctorale N°512 InfoMaths
Spécialité : Informatique

N° d'ordre NNT : 2022ECDL0029

Présentée et soutenue le 14/12/2022 par :
Thomas DUBOUDIN

Deep Diversity Learning for Better Generalization to Unseen Domains

---

Directeur de thèse : **Pr. Liming Chen**
Co-directeur de thèse : **Dr. Emmanuel Dellandréa**

Jury

Mme. Nicole Vincent	Professeur, Université Paris-Cité	Rapporteuse
M. Amaury Habrard	Professeur, Université Jean-Monnet	Rapporteur
Mme. Su Ruan	Professeur, Université de Rouen	Examinatrice, Présidente
M. Liming Chen	Professeur, ECL	Directeur de thèse
M. Emmanuel Dellandréa	Maître de conférences HDR, ECL	Co-directeur de thèse
M. Gilles Hénaff	Responsable STI, Thales LAS France	Invité
M. Corentin Abgrall	Ingénieur IA, Thales LAS France	Invité

ABSTRACT

A growing number of embedded applications, confronted with diversified, shifting, and uncontrolled environments, require an increased degree of adaptability and analysis capabilities to fulfill their task. Preprogrammed actions are no longer able to deal with these new sets of tasks and are therefore being replaced by a promising paradigm: deep learning.

However, deep neural networks are susceptible to data distribution shifts occurring between training and use. This apparent flaw prevents the widespread deployment of deep networks in embedded products. Furthermore, it is impossible to gather and add enough data to the training set to cover all possible shifts due to their tremendous diversity.

The origin of this vulnerability lies partly in the shortcut-learning behaviour of deep networks: they learn only the most efficient patterns, no matter how spurious, and completely disregard the others. Confronted with a new distribution in which the predictive patterns are partially different, a network that learned a limited subset of features would be less likely to be able to make a proper decision.

In collaboration with Thales Land and Air Systems, this work therefore aims to develop solutions to mitigate the domain shifts performance drop in deep networks. This work has two main contributions.

Firstly, we propose a new deep generative architecture that mitigates the shortcut-learning behavior in an under-explored setting. Previous state-of-the-art works relied explicitly on shortcut-contrary samples and increased their importance in the training procedure. In this work, we demonstrated on several different synthetic benchmarks that such particular samples were not needed for shortcut avoidance and further confirmed the effectiveness of our approach on a realistic benchmark.

Secondly, the work presented here focuses on more general and realistic domain shift situations in which only a single domain is available during training. Test-time adaptation to the encountered data has emerged as a promising set of strategies to efficiently increase performance when facing new domains at use time. They, however, rely on a model trained with the standard procedure, which, as previously stated, will ignore some predictive patterns. We propose a training-time approach complementary to test-time adaptation. Our

method seeks to learn both the patterns learned through standard training and the normally "hidden" ones, enabling a more thorough test-time adaptation. Based on extensive experiments, we show that our approach improves the quality of predictions on domains unseen at training-time.

Keywords: deep learning, domain generalization, out-of-distribution generalization, test-time adaptation, shortcut-learning, bias avoidance, covariance shifts.

RÉSUMÉ

De plus en plus d'applications embarquées, confrontées à des environnements diversifiés, changeants et non-controllés, nécessitent un très haut degré d'adaptabilité et de fortes capacités d'analyse pour mener leur tâche à bien. La pré-programmation d'actions n'est plus suffisante pour effectuer ces nouveaux types de tâches, et est donc en train d'être remplacée par un paradigme prometteur : l'apprentissage profond.

Cependant, les réseaux de neurones sont vulnérables à des changements de distribution (ou domaine) de données entre l'apprentissage et l'utilisation. Ce défaut apparent empêche un déploiement fréquent dans des produits embarqués. De plus, il est impossible de récolter et d'ajouter au jeu de données d'entraînement suffisamment de données pour prévenir tous les changements de distribution possibles, à cause de leur importante diversité.

L'origine de cette vulnérabilité se trouve en partie dans le comportement d'apprentissage de raccourcis des réseaux de neurones profonds: seulement les motifs prédictifs les plus efficaces, aussi fallacieux soient-ils, sont appris, les autres sont fortement ignorés. Confronté à une nouvelle distribution, dans laquelle les motifs prédictifs sont partiellement différents, un réseau qui aura appris un ensemble limité de caractéristiques sera statistiquement moins capable de prendre une décision correcte.

En collaboration avec Thales Land and Air Systems, l'objectif de ce travail est donc de développer des solutions pour atténuer la chute de performance des réseaux de neurones lors d'un changement de domaine. Ce travail comprend deux contributions principales.

Premièrement, nous proposons une nouvelle architecture générative de réseaux de neurones qui permet de limiter l'ampleur de l'apprentissage de raccourcis dans un contexte sous-exploré. Les précédents travaux de l'état de l'art reposaient explicitement sur des éléments particuliers, dont les raccourcis ne sont pas alignés avec la majorité des autres données, et augmentaient leur importance dans la procédure d'apprentissage. Dans cette contribution, nous avons montré que ces éléments n'étaient pas nécessaires pour éviter les raccourcis, grâce à des jeux de données synthétiques construits dans cet objectif, et avons de plus validé l'efficacité de notre approche sur un

jeu de données réaliste.

Dans un deuxième temps, le travail présenté ici se concentre sur des situations de changement de domaines plus générales, dans lesquelles un unique domaine est disponible pendant l'apprentissage. L'adaptation au moment du test aux données a emergé comme un ensemble de stratégies prometteur pour efficacement améliorer les performances quand de nouveaux domaines sont rencontrés lors de l'utilisation. Cependant, ces méthodes s'appuient sur des modèles entraînés avec la procédure standard, qui ne permet pas l'apprentissage de la totalité des motifs prédictifs, comme nous l'avons vu. Nous proposons donc une modification de la procédure d'entraînement complémentaire avec l'adaptation au moment du test. Notre méthode permet l'apprentissage des motifs naturellement appris ainsi que des motifs habituellement ignorés, et par conséquent, permet une adaptation plus approfondie. Sur la base de plusieurs expériences, nous montrons que notre approche améliore la qualité des prédictions sur des domaines jamais rencontrés au moment de l'apprentissage.

Mots-clés: apprentissage profond, généralisation de domaine, adaptation au moment du test, apprentissage des raccourcis, apprentissage débiaisé.

REMERCIEMENTS

Je tiens à remercier Liming Chen et Emmanuel Dellandréa, mes directeurs de thèse, pour leur investissement dans mes travaux, pour les conseils et remarques prodigués lors de la rencontre de difficultés, et pour leur aide cruciale dans l'écriture des articles.

Je remercie la société Thales Land and Air Systems, et plus particulièrement Gilles Hénaff, responsable du pôle intelligence artificielle du service OME, pour m'avoir permis de mener ces travaux de recherche dans le cadre de cette thèse CIFRE.

Merci à tous mes collègues de Thales et du LIRIS pour leur soutien et leur aide. Merci particulièrement à Corentin, membre du pôle IA de Thales, pour nos nombreuses interactions, pour son investissement et son intérêt dans mes travaux. Merci à Rémy, Thomas et François, doctorants du pôle IA, pour nos échanges instructifs. Merci enfin à mes collègues du LIRIS : Maxime, Amaury, Dimitri, Hugues, Nicolas, et Nicolas, pour avoir fait des journées au laboratoire un moment plaisant, et pour avoir été toujours prêts à échanger.

Merci à mes parents qui m'ont soutenu tout au long de ce travail, sur le plan personnel et professionnel, et se sont intéressés à mes travaux, ainsi qu'au reste de ma famille. Je remercie du fond du coeur ma compagne, Marion, pour m'avoir supporté patiemment pendant les périodes chargées, et pour ses encouragements infatiguables. Je tiens aussi à remercier mes amis, grâce à qui cette période de ma vie ne se résume pas à la thèse.

Finalement, merci aux membres du jury d'avoir accepté la tâche d'évaluer les travaux réalisés dans le cadre de cette thèse.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 CONTEXT

Thales Land and Air Systems (LAS) develops several embedded computer vision applications for civilian or defense-related purposes. These applications require a high level of adaptability and analysis capabilities, as they are designed for tasks currently performed by human operators, and for which expert training specific to each task is furthermore required. Thales LAS's main interest in automating these tasks is to speed up their implementation. These image analysis tasks cannot easily be automated with traditional approaches because they require a high level of understanding of an uncontrolled and very diversified environment to be completed. An explicit programming based on visual cues thus cannot reach a satisfying performance on all encountered situations.

Machine learning approaches, and particularly deep learning ones, have emerged as a very promising paradigm to deal with these kinds of tasks. Instead of explicitly programming the decision process of the application, which requires careful planning of numerous possible scenarios, it is learned based on a large amount of annotated data, with limited human intervention in the loop. Deep neural networks have thus reached human-level performance in several computer vision tasks that are of interest to Thales LAS, such as image classification, objects detection or semantic segmentation [44, 46, 68], and are now among the preferred strategy for computer vision tasks.

However, deep neural networks suffer from a dramatic shortcoming when the images encountered at use-time differ from the images used for training: they are susceptible to data distribution, or domain, shifts between training and use [7, 23, 51]. Examples of such different (and yet dealing with the same semantic concepts) data domains are given in figure 1.1. This flaw leads to a use-time performance drop which can be large even if, to the human eye, training and use images are close. When possible, more data can be gathered from diverse sources and added to the training set to create a larger data distribution, but as the diversity of possible shifts is tremendous, this will not enable robust predictions for all possible situations. This flaw seriously limits the widespread deployment of deep networks in embedded products. It is usually dealt with by first detecting a domain shift and then returning control to the human operator.

Figure 1.1: Samples of the Office-Home domain adaptation and generalization benchmark. Different data domains are shown in the vertical axis, and different classes in the horizontal ones.

Our overall goal is to enable the continued operation of the embedded application by the deep learning model, even when confronted with a distribution shift, by maintaining the performance drop below a threshold that would force human intervention.

1.2 SPECIFIC PROBLEMS AND OBJECTIVES

The origin of the domain shifts vulnerability lies in several factors. First, a moderate intrinsic bias towards textures rather than shapes, linked to the use of cropping in data augmentation and to the convolutional architectures themselves, harms robustness when the domain shift encountered is essentially a texture shift [48, 49, 101]. Then, depending on the magnitude of the distribution shift, the predictive patterns on the use-time distribution might simply be different from those learned on the training distribution. In our works, we focus on a third factor called shortcut-learning: deep networks rely exclusively on the simplest and most predictive patterns for the task at hand, no matter how spurious they are, without taking into consideration the less effective yet still predictive patterns. A classic given example is the binary camel or cow classification problem: in most of the training images, cows will be on a grassy background while camel will be on a desert background. A neural network will thus learn that green grass is a pattern predictive of the cow class and that sandy desert is predic-

tive of the camel class, and these patterns will be stronger predictors of the class than the true, causal, animal patterns. This behavior hinders the out-of-domain generalization ability: the limited subset of learned patterns may be partially missing (or showing the use-time images but decorrelated from the labels or correlated with a different label than in the training set) from the target domain, especially if they are spurious, and a neural network relying exclusively on them will be unable to make a proper prediction. Enlarging the set of learned predictive patterns appears to be a promising lead to mitigate, at least in part, the domain shift performance drop.

Quantifying the diversity of learned patterns is a difficult task, as simple metrics are often unable to measure the true amount of semantically different patterns that are effectively useful for the task at hand. Likewise, explainability methods do not allow a precise enough diagnosis and require a high amount of human intervention. Furthermore, the performance gain linked to the use of a diversity-seeking approach in a real-life domain shift situation cannot be used as is, as most of the time, a domain shift cannot be reduced to a simple missing patterns issue. Our first goal is thus to measure precisely a training procedure's effectiveness in finding the "hidden" patterns and propose an approach able to reach a satisfying performance if the existing baselines fail to.

Our second and final goal is to deal with more general domain shifts. Learning the "hidden" patterns alone will most likely not result in a significant improvement because of the many possible label decorrelation or anti-correlation between training and test patterns. Test-time adaptation, however, enables an adaptive modulation of patterns specific to a particular test domain and has proved effective in this context [121, 123]. However, test-time adaptation methods rely on a model trained with the standard training procedure which misses a lot of possible prediction cues. Combining both approaches should enable a more thorough adaptation and reach greater improvements. We further focus on the single-source setting, in which only a single data domain is available during training. This setting is more realistic and practical than its multi-source counterpart: the systematic availability of several different and identified data domains is quite a generous assumption, especially in defense or healthcare-related fields where the sharing of data between organizations is heavily legally regulated.

1.3 CONTRIBUTIONS

1.3.1 *Learning "hidden" patterns*

- Our first contribution to the study of the shortcut-learning behavior is the public release of a set of synthetic datasets [27]. They rely on the popular MNIST [69] and CIFAR-10 [65] datasets. They were specifically crafted with two main goals in mind: first, so that the accuracy of a model on the test set reflects how well the less effective predictive patterns are learned, and second, to prevent a reliance on unrealistic and impractical hypotheses that are often used in the field. As a result, even though they rely on the simple and already mastered MNIST and CIFAR-10, no existing shortcut avoidance baselines perform better than the standard training procedure on these datasets. We constructed these datasets by adding a classification shortcut (or bias) to the original images in the shape of visual cues absolutely correlated with the labels. The shortcuts were chosen to be of different types for each dataset: spatially mingled with the causal patterns, spatially located in a narrow area, color-based or location-based. Unlike all previous synthetic voluntarily biased datasets, the correlation between shortcuts and labels is absolute (*i.e.* there are no samples exhibiting the shortcut of another class). We do so to investigate whether or not it is possible to learn patterns semantically different from the shortcuts without relying explicitly on a few shortcut-contrary samples, as almost all existing works do. We argue that assuming the systematic existence of such samples and increasing their importance in the training procedure limits both the applicability of a method and its performance. First there will be no shortcut-contrary samples for certain unusual tasks or when using synthetic data. Then, increasing the importance of samples behaving differently from the majority may lead to increase the importance of labeling errors, with a counter-productive effect on the resulting accuracy. Finally, in most of the situations, the underlying causal patterns are also showing alongside the shortcut on the images, they are simply less predictive. Learning them using the whole training set will provide better generalization capabilities than only using a small subset. Samples of the proposed datasets are available in figure 1.2.

- As no baselines yield satisfying results, our next contribution is a method able to succeed in finding the "hidden" patterns on all of our benchmarks. Our approach relies on a simple idea: if one can somehow "remove" the classification shortcuts from an image, training on these images will enable the learning of the previously "hidden" patterns as all the more efficient patterns

Figure 1.2: Samples of our proposed benchmarks. Even rows: training set, following odd rows: corresponding test set.

are now gone. We propose to implement this idea using a disentangling auto-encoder that separates the information contained in an image between what is naturally used by a classifier and what is not. Our architecture makes use of a supplementary adversarial entropy maximization loss on the ground that high entropy images are less likely to contain strong prediction clues for a network. We finally further confirm the effectiveness of our approach on a realistic benchmark: the Biased Action Recognition dataset (BAR) [86] and show that it reaches state-of-the-art performance.

1.3.2 *General test-time domain shifts*

Our disentangling auto-encoder suffers from several drawbacks. First, it is only designed to learn both the shortcuts and the underlying patterns. Indeed, ignoring the shortcuts would require to know in advance that they are spurious and can be safely ignored without losing causal information, or, said differently, that the training dataset is biased and this is an often unrealistic hypothesis. Then, it is also not equipped to deal with possible decorrelation (or anti-correlation) between training and test patterns or more general domain shifts of the covariate shifts family. Secondly, it is computationally heavy (as it requires two classifiers, an encoder, and a decoder, all simultaneously trained), which prevents a really widespread use in an industrial context. To deal with general domain shifts in a more efficient fashion, our main contribution is a novel modification of the training procedure, in the shape of a slight architecture change, compatible with a wide range of networks, and an additional training loss. It is furthermore

meant to be used conjointly with a lightweight test-time adaptation method. Compared to existing works, our approach is thus simpler, more stable during training, and rely on very few hypotheses. We again aim to find predictive patterns that are semantically different from the normally learned ones and propose to do so with a generalization minimization constraint, or shortcut avoidance loss. The simplicity, or inductive, bias [35, 57] of deep networks promotes the learning of the simplest and most generalizable patterns (on the training distribution). As a result, by looking for patterns that are both predictive and generalize poorly, we are able to learn precisely those normally ignored. We do so by adding a secondary prediction branch to the original architecture and by training it with an additional shortcut avoidance loss that promotes the learning of samples' specific patterns, *e.g.* patterns that lower the loss on a specific batch of data but with a limited effect on the others. The primary branch is trained normally and thus tasked with learning the original patterns, while the secondary one is tasked to seek the less effective ones. Single-source experiments on the classic domain generalization benchmarks PACS [72], and Office-Home [113] confirmed the effectiveness of our approach against several state-of-the-art single-source baselines.

1.4 PUBLICATIONS

Based on the work done during this thesis, two papers have been published in international conferences and workshops, and one is currently under review.

- [27] Thomas Duboudin, Emmanuel Dellandréa, Corentin Abgrall, Gilles Hénaff, and Liming Chen. "Encouraging Intra-Class Diversity Through a Reverse Contrastive Loss for Single-Source Domain Generalization". Published in the *2021 Adversarial Robustness in the Real World (AROW) workshop of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

  This early work proposed the aforementioned MNIST-based dataset with color-based shortcuts and a novel method to learn the digit's shapes despite the highly predictive colorization. Our method relied on a reversed contrastive loss that pushed for large intra-class clusters and small boundaries between classes in the latent space of the network. Unfortunately, this approach did not transfer to the CIFAR-10-based datasets and was, as a result, discontinued. Nonetheless, this laid the foundations and ideas for future works.

- [28] Thomas Duboudin, Emmanuel Dellandréa, Corentin Abgrall, Gilles Hénaff, and Liming Chen."Look Beyond Bias with Entropic Adversarial Data Augmentation". Published in the *2022 International Conference on Pattern Recognition (ICPR)*.

This work introduced the three synthetic benchmarks and the auto-encoder-based architecture able to succeed on them and on the BAR benchmark.

- [29] Thomas Duboudin, Emmanuel Dellandréa, Corentin Abgrall, Gilles Hénaff, and Liming Chen. "Learning Less Generalizable Patterns for Better Test-Time Adaptation". Published in the *2023 International Conference on Computer Vision Theory and Applications (VISAPP)*.

  This work introduced the final contribution of the thesis, a novel domain generalization method that complements test-time adaptation, and its results on the PACS and Office-Home datasets that improve over the state-of-the-art.

1.5 CONTENT

This thesis is organized as follows:

In Chapter 2, we review the state-of-the-art on the domain generalization problem and its derivatives: shortcut-learning mitigation, biases avoidance, and test-time adaptation.

In Chapter 3, we present our work on shortcut avoidance, comprising of a detailed description of the synthetic benchmarks and of the details of our auto-encoder architecture. We show its performance against several state-of-the-art baselines, for different deep architectures, on both our benchmarks and the BAR benchmark.

In Chapter 4, we detail our final contribution and its benefits to a test-time adaptation method in the context of general domain shift situations. We extensively confirm its effectiveness against a diversified panel of state-of-the-art baselines in the single-source setting.

In Chapter 5, we summarize our work and contributions and lay out some broad directions and unanswered questions for future improvements in domain generalization.

2

STATE-OF-THE-ART

In this chapter, we review existing state-of-the-art works in the research field of out-of-distribution generalization and in its corollaries. In section 2.1, we first review theoretical and experimental studies exploring the origin of the domain shifts vulnerability. In section 2.2, we review existing debiasing methods, as biases sensitivity is an interesting and informative sub-case of general domain shifts situations. In section 2.3, we review training-time methods designed to deal with general domain shifts and their limitations. Finally, in section 2.4, we review algorithms belonging to the new test-time adaptation paradigm. We conclude this chapter in section 2.5.

2.1 ON THE ORIGINS OF DOMAIN SHIFTS VULNERABILITY

Seemingly contrary to humans, deep networks are highly susceptible to shifts between training and test domains. Explaining this vulnerability is still an ongoing work, but several quirks have already been identified in both the family of deep models and the training procedure. First, a proper definition of the domain shifts is required to explain why networks fail to maintain their accuracy when encountering one. A domain shift is a change of data probability distribution between a model's training and its use. For a computer vision task, in the usual supervised-learning setting, it means that the joint distribution over images x and labels y differ between training and test:

$$p_{train}(x, y) \neq p_{test}(x, y) \tag{2.1}$$

This broad definition allows for every possible shift, including some which we do not expect or even want a model to be robust to (most notably if the sets of training and test labels are disjointed). It thus must be narrowed down to specific cases to be useful. In our works, we focus on the covariate shift [95]. It is a particular kind of shift for which it is reasonable to expect more performance robustness from a network. It is defined as follows:

$$p_{train}(x) \neq p_{test}(x), p_{train}(y|x) = p_{test}(y|x) \tag{2.2}$$

In a covariate shift, the theoretical functional relationship between images and labels stays the same during training and at test-time. For instance, it happens in face recognition tasks when the ethnicity of faces encountered at use-time differs from the ones of training faces or in histopathology-related tasks when different colorization protocols are used during training and use. Note that depending on the magnitude of the covariate shift, it will not be possible, even theoretically, to recover all the accuracy lost on the target domain. For instance, consider a binary classification problem with only two kinds of shapes in the training data: crosses (class 0) and circles (class 1). At test-time, the shapes encountered are crosses (0) and squares (1). As the model did not encounter any squares during training, its decision for squares will be uncertain, towards either 0 (crosses) or 1 (circles), depending on its intrinsic biases and the learned patterns.

The main challenge of covariate shift thus lies in learning patterns predictive of the correct labels on the test domain while having no information about it at training-time (in a domain generalization setting, contrary to a domain adaptation setting, no information about the target domain is available at training-time). Real-life covariate shifts are not as extreme as the example above: some predictive patterns in the training set are always potentially useful on the test set, provided that they are indeed effectively learned at training-time. For our own work, we assume that the encountered distribution gap is narrow enough to be entirely bridged by a theoretical algorithm or, said differently, that the training set contains enough information to yield a model robust on the new test distribution. However, numerous experiments show that deep neural networks favor particular predictive patterns over some others and can even completely ignore some of them.

2.1.1 *The texture bias of deep neural networks*

Trained deep networks favor texture-based patterns over shape-based patterns. Geirhos *et al.* [39] indeed showed that a ResNet [46] trained on ImageNet [25] performs poorly and relies most of the time on textures rather than shapes. They demonstrated it by evaluating their network on an ImageNet test set in which textures and shapes are decorrelated trough the use of a style transfer method [37, 55]. An example of an image with such conflicting texture and shape cues is shown in figure 2.1, alongside its corresponding network prediction. As textures are more likely to be spurious than shapes and can change wildly between domains [84], an over-reliance on textures harms the out-of-distribution generalization capabilities of the networks. Many usual domain shifts are indeed essentially texture shifts: for instance, a model trained on synthetic data and then deployed in the wild, a test-time change in lighting and weather conditions, or a

(a) Texture image
81.4% **Indian elephant**
10.3% indri
8.2% black swan

(b) Content image
71.1% **tabby cat**
17.3% grey fox
3.3% Siamese cat

(c) Texture-shape cue conflict
63.9% **Indian elephant**
26.4% indri
9.6% black swan

Figure 2.1: Samples of the conflicting cues images and their corresponding network prediction. (a) Texture image without shape content. (b) Original ImageNet image. The predictions for both images are correct. (c) A hybrid image, exhibiting a cat shape and an elephant texture, obtained using a style transfer algorithm. The network still predicts the elephant class, even though it is in a less confident fashion than originally, which indicates a particular reliance on textures.

change of sensor between training and use data. The origin of this bias lies in several factors. First, deep convolutional networks, which are very commonly used in computer vision tasks, seem to possess an intrinsic bias towards colors and textures as Scimeca *et al.* [101], and Hermann and Lampinen [49] showed on synthetic datasets: facing equally predictive cues of different kinds (color, texture, shape), deep convolutional neural networks prefer colors and textures over shapes. Secondly, the use of large image crops as data augmentation also has an adverse effect on the learning of shapes: Hermann *et al.* [48] showed that simply removing crops from the set of data augmentations noticeably improved the performance on the ImageNet test set with conflicting cues from [39]. These findings have given rise to numerous domain generalization works relying on style transfer to alleviate the texture bias, which we will review in section 2.3.

2.1.2 *The shortcut-learning phenomenon*

Alongside this preference for textures, deep networks suffer from an additional drawback: confronted with a large set of roughly equally predictive patterns in the training set, deep networks will only learn a small subset of them. This behavior is related to the textures bias, as the only patterns learned are often the texture-based ones, but is ultimately distinct as it manifests itself even when all available patterns are shape-based [38, 101]. An illustration of this flaw, originating from the work of Pezeshki *et al.* [93], is available in figure 2.2. Learning only a narrow set of predictive patterns increases the risk of falling short of cues at test-time if a learned pattern becomes missing and is thus detrimental to the out-of-distribution robustness. This behavior has been experimentally studied on synthetic benchmarks, in [49, 101],

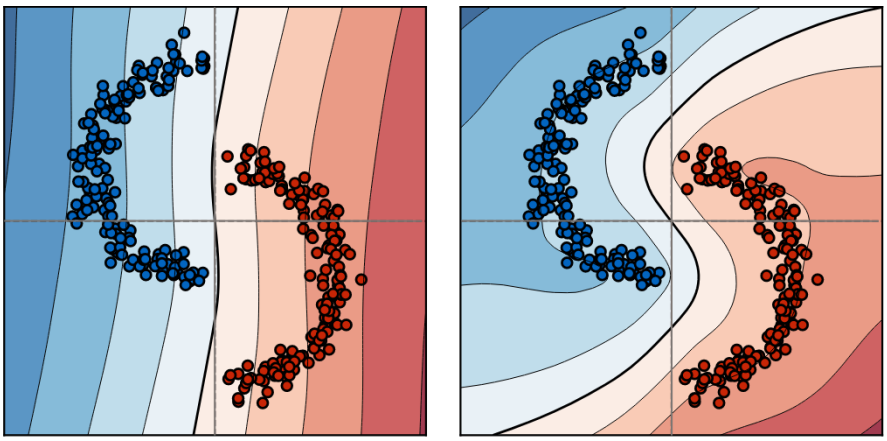

Figure 2.2: Illustration of the shortcut-learning behavior on a vertically linearly separable double crescent classification task. On the left: the prediction boundary learned by a network trained with the standard procedure. Note that, despite being correctly predictive of the labels in $\sim 70\%$ of the samples, the y-axis is almost entirely ignored by the network. On the right: the desired behavior. The y-axis is taken into account.

but theoretical explanations remain sparse and scarce. Pezeshki *et al.* [93] point out the responsibility of the stochastic gradient descent in this phenomenon. Because of a coupled training dynamic, an increase in the strength of a particular feature has a detrimental effect on the learning of other coupled features. This phenomenon is also related to the simplicity bias of deep networks, which promotes the learning of the "simplest" solutions (in the sense of low-rank solutions) [57] and is thought to explain why deep networks generalize well if the test images are coming from the same distribution as the training images. Harshay *et al.* [104] explored the pitfalls of the simplicity bias and blamed it for the shortcut-learning behavior. It can also be responsible for the lack of reliable confidence estimates and can even hurt in-domain generalization as the simplest patterns might be learned even though more predictive but complex ones exist.

2.1.3 *The numerous biases of training datasets*

All these phenomena are common occurrences because of the fairly frequent presence of shortcuts or biases in the training datasets [32, 109]. Biases are shortcuts that are moreover spurious, and as such are more likely to be missing at test-time because they are entirely determined by biases in the data gathering procedure. These biases and shortcuts arise because of data gathering biases, such as relying on too few or particular sources of data (as in [23]), or because of non-stationary environments that make certain patterns become spurious over temporal or geographical changes (as in [7]). Note that not all

shortcuts are spurious. For instance, in a car brand classification task from car pictures, the brand logo located on the front or the back of a car will be a non-spurious, *i.e.* causal, shortcut if a large proportion of training images exhibit it. Despite this shortcut being causal, we nonetheless wish to learn other semantically different cues.

2.2 DEBIASING AS ROBUSTNESS TO SPECIFIC DOMAIN SHIFTS

2.2.1 *Methods*

Several works are dedicated to preventing the learning of dataset biases by deep neural networks. Some of them do so to ensure a better out-of-distribution robustness [62, 70, 86] and others to ensure the fairness of deep learning algorithms regarding sensitive data attributes, such as religion, skin color, or gender [60]. Such debiasing methods are essentially determined by the amount of information assumed to be available about the biases. There are roughly three situations:

- A first line of works requires the bias to be given as an auxiliary label. For instance, Kim *et al.* [60], are able to make a network invariant to a known bias using a gradient reversal layer, in a similar fashion to early domain adaptation works [36, 111], that forces the features embedding to unlearn the biases. An additional classifier layer is added to the original architecture, plugged after the features extractor, and tasked with predicting the bias from an image. Simultaneously, the features extractor is trained to maximize the bias prediction layer loss. This approach suffers from an important practicality drawback: requiring the biases to be given as auxiliary labels is tedious from a human annotation perspective and restricts the application of the method to biases that can be spotted with the human eye.

- A newer line of works rely upon a slightly weaker hypothesis. It is only required to know in advance that a given dataset is strongly biased. What is learned naturally by a network is then assumed to be only biases and, as such, can be safely ignored without losing causal information. Most of the methods [1, 22, 62, 70, 76, 86] focus on finding counter-examples, or bias-contrary samples, *i.e.* samples that do not share the majority bias of their class, often via a loss-based criterion. Once such samples are found, their importance during the training is increased via an over-sampling scheme as in Just-Train-Twice (JTT) [76] or via a loss re-weighting scheme as in Learning-from-Failure (LfF) [86]. As a result, the biases that would be otherwise learned become less efficient than the causal predictive patterns and are thus ignored to the benefit of the latters. Closest to our first contribution is the work of Kim *et al.* [62] that allows the creation

of a debiased dataset from a biased one by generating hybrid images in which biases and labels are decorrelated. A model is then simply trained on this decorrelated dataset and does not learn the biases as they are no longer predictive. Iterating over this idea, in [70] (LDD), Lee *et al.* propose to train on "virtual" hybrids by disentangling bias and causal patterns at features level and creating combinations unseen in the training set before training on them.

We argue that it is desirable to design methods that do not rely on such few portions of samples. In real-life situations, it could lead to overestimating the importance of outliers that should be ignored, such as annotation errors (Northcutt *et al.* [89] estimated that the average annotation errors percentage in usual computer vision datasets was 3.3%). It may also be optimistic to find counter-examples of the biases in some situations, *e.g.* with synthetic datasets where there is often a low diversity of situations. Furthermore, most of the time, the causal patterns are still present in the biased images: they are simply "hidden" behind the more effective biases. Finding the causal patterns in the biased samples should therefore be possible and probably leads to better generalization than bias-contrary-based debiasing methods due to the reliance on a vastly larger amount of data. Besides, knowing in advance that a dataset is biased is often an unrealistic hypothesis (spurious biases can be noticeable background or context, as in [7], but also inconspicuous hospital-specific clues, as in [23]).

- A final line of works, fairly succinct, makes no hypotheses and proposes to learn both the patterns that would be learned naturally and the less effective "hidden" ones (no matter if spurious or not, as it is not something that can always be easily inferred from looking at the data). Our first algorithmic contribution falls into this category. The only other work, to our knowledge, is the spectral decoupling regularization from Pezeshki *et al.* [93] that encourages the learning of new features rather than the refining of already existing features by penalizing the raw prediction's confidence.

2.2.2 *Benchmarks*

Several datasets are publicly available to benchmark debiasing methods. For a precise measure of the effect of a debiasing method, many synthetic datasets were created, in which the bias is completely controlled. Often, such datasets are based on the classical MNIST [69] and CIFAR-10 datasets [65]. Arjovsky *et al.* [3] first proposed an MNIST-

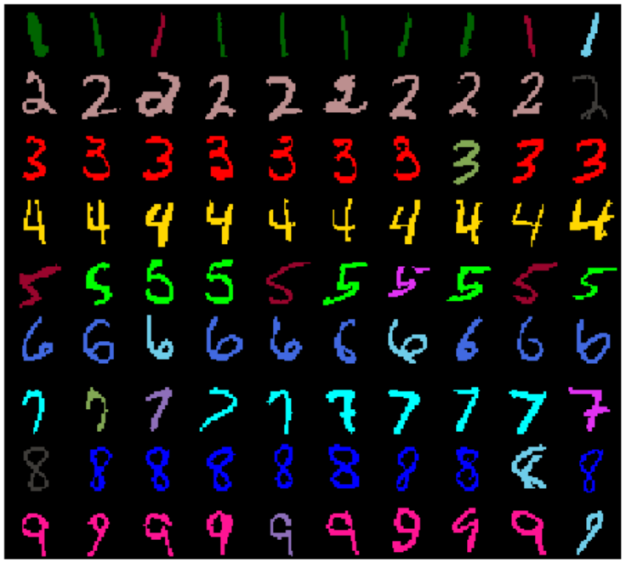

Figure 2.3: Samples of the colorized MNIST-based dataset proposed in [1]. Some samples are colorized differently from the majority of samples of the same class, and are, as such, called counter-examples, bias-contrary or shortcut-contrary samples.

based two-classes dataset with label noise and two different source domains with different strengths of color-label correlation, while in the target domain, colors and labels are almost completely decorrelated. Pezeshki *et al.* [93] propose a slightly more complex version of this benchmark by removing a source domain from the training set, thus working with a single training domain. Faruk *et al.* [1] produce a more complex benchmark by using the original ten MNIST classes and by coloring each class with its corresponding color in most, but not all, of the cases (the training set is illustrated in figure 2.3). Two different target sets are available: a systematic shift set, in which the same colors as in training are used but strongly correlated with different labels, and a non-systematic shift in which the colors used for the target set colorization are entirely different from the training colors. Dagaev *et al.* [22] rely instead on the more complex CIFAR-10 dataset and bias the images by adding a small colored patch, in which the color is again spuriously correlated with the labels, or by adding a low magnitude random mask the size of the image different for every class.

Regarding the real-life datasets, the most used ones are the BAR (Biased-Action-Recognition) dataset [86] and the celebrity faces dataset CelebA [79]. The BAR dataset consists in 1941 training images and 654 test images falling into six action categories. A sample of the images is available in figure 2.4. Training data exhibits a heavy background bias related to the action: 'climbing' is often illustrated with grey, rocky, background, 'throwing' with a recognizable baseball field background, *etc*. Test data, however, does not, as it exhibits a higher diversity of

Figure 2.4: Illustration of the BAR dataset. The bias-contrary samples (minority in training and validation data, and majority in the test data) are circled with red.

backgrounds and environments and as such, the standard training procedure yields a poorly performing model on the test data. The CelebA dataset is a multi-task dataset comprising of 162k images with a set of binary attributes associated with each image. To study debiasing, it is often used in a binary blond or not hair-color prediction task as the hair color is spuriously correlated with the gender (the blond men group represents only 0.85% of the training data: 1387 out of 162k examples).

2.3 DOMAIN GENERALIZATION

2.3.1 *Robustness through multi-source training*

Most of the domain generalization methods assume to have access to data coming from several identified domains during training. They aim to learn more robust features by seeking features that are shared among all source domains, on the ground that patterns shared among several domains are more likely to be found in a new, unseen, domain. This domain invariance can be achieved through a great diversity of practical approaches:

- Many recent works use adversarial strategies: in the work of Tzeng *et al.* [111], the features extracted by a features extractor are fed to a discriminator tasked to find the original domain of the image. The discriminator is trained to minimize the domain classification error, while the feature extractor is trained

to maximize it, hence making the features indistinguishable between domains. This family of works was initially developed in the context of domain adaptation but can be adapted in the multi-source domain generalization setting by employing all the domains at disposal during training, *e.g.*, [14], and [74].

- Uniquely, the work of Krueger *et al.* [66] is enforcing invariance at loss level by minimizing the variance of the loss over all training domains alongside the usual average of the loss per batch, following the idea that a similar loss value among all domains is indicative of similar patterns learned.

- Some works rely on meta-learning, of a similar kind than MAML [34], and simulate a train-test domain shift during training using the different source domains. An additional meta-objective forces the gradient update to minimize both the training and the virtual test loss. Li *et al.* [73] introduced the main algorithm, and Balaji *et al.* [5] refined it with technical improvements in the meta-optimization steps.

- Self-supervised strategies have also emerged to solve the problem of multi-source domain generalization. Carlucci *et al.* [13] used a self-supervised strategy based on solving jigsaw puzzles: the images are divided into tiles that are shuffled before being given to the classifier. The classifier has two goals: classify the samples and find the shuffling permutation. By having a supplementary objective not related in any way to the classification, the network will learn other patterns than those strictly sufficient to the classification task and possibly generalize better to a new distribution. Recent self-supervised strategy [15, 18, 19] that yield a better pretraining than the usual supervised ImageNet one were found to perform poorly in domain shifts situations [61, 87]. More recently, however, a fully self-supervised approach has for the first time reached results equivalent or better, in some cases, than the usual ImageNet pretraining in domain shift situations [127].

- Batch-normalization [58] has also sparked interest in the context of domain generalization. Seo *et al.* [103] proposed a relatively lightweight architecture modification: a batch-normalization layer is used for each training domain independently (more specifically, the affine parameters are domain specific) to avoid internal covariate shifts if several source domains are used at the same time. The statistics used for normalization are a mix of global and domain-specific statistics. Segu *et al.* [102] proposed to keep only the domain-specific normalization statistics and use the same affine parameters for all domains. Both approaches require a merging strategy at test-time: the prediction for a test-

time sample is a weighted average of the outputs obtained with all the different statistics or affine parameters.

- Arjovsky *et al.* in [3] proposed Invariant Risk Minimization (IRM): a more principled approach to learn invariant features. After an analysis of the causes of out-of-distribution generalization failures, they propose to learn a data representation so that the optimal classifier on top of this representation matches for all source domains. Through a relaxation, this objective boils down to minimizing both the original loss and an additional gradient norm regularization. Their work spawned many subsequent derivatives, such as in [2], in which an information bottleneck constraint is added to the existing Invariant Risk Minimization framework.

- A final noticeable family of works relies upon gradient "surgery" to learn invariant features, on the basis that disagreements between gradient's directions of different source domains are indicative of the learning of domain-specific features and hamper out-of-distribution generalization. Mansilla *et al.* [81], and Parascandolo *et al.* [90] concurrently proposed a gradient dropout strategy that keeps only the coefficients whose signs match over all (or a simple majority) source domains (a strategy known as AND-masking). Shahtalebi *et al.* [105] analyzed the remaining failure situations. Their work concluded that dead zones (weights sub-space for which gradients systematically disagree and where weights, as a result, are never updated) and high sensitivity to noisy gradients (because of the strict threshold zeroing strategy) have a strong responsibility in these failure situations. Thus, they proposed a relaxed strategy, dubbed smooth-AND-masking, that more gradually promotes agreement.

Important methodological points in domain generalization works are often overlooked, which resulted in the 2019 DomainBed [42] catastrophe. The most important methodological matters are the model and the hyper-parameters selection. In a standard setting, the test model is the one that performs best on a left-aside validation set. This superior performance will be reflected in the test set, as both datasets come from the same data distribution (and as deep neural networks are not very prone to over-fitting [8]). Likewise, hyper-parameters are searched using the validation set and transfer well to the test set. In a domain shift situation, one could use the target domain to select the final model or the hyper-parameters (the 'oracle' selection strategy), but this is not realistic: in a real-life scenario, samples of the target domain are not available (if they are, the situation is a simpler domain adaptation matter). As a result, a validation set coming from the training domain is used most of the time (when an indication of the methodology used is provided, which is far from systematic).

This validation set strategy is sub-optimal, as the best test accuracy is not necessarily reached for the best validation accuracy. Very few works have proposed alternative selection criteria. Ye *et al.* proposed in [122] to select a model with both high validation accuracy and low class-wise variation between features of different source domains.

Following these observations, in 2019, Gulrajani and Lopez-Paz [42] analyzed a lot of domain generalization algorithms and benchmarks "in search of lost domain generalization". They detailed ways to carefully and realistically evaluate multi-source domain generalization methods using larger models, stronger data augmentations, correct models, and hyper-parameters selection (without any use of the target domain), and doing all on more datasets. All their contributions resulted in the DomainBed benchmark (`https://github.com/facebookresearch/DomainBed`). They concluded that no methods were significantly above the basic expected risk minimization procedure (ERM), where data from all domains is merged into one single dataset with the network trained on it. Their findings are corroborated by Cha *et al.* [16] who look into the latent spaces from differently trained networks, one with DANN [36] and another with the basic ERM. They have shown that ERM naturally tends to yield domain-invariant features. Methods published after the release of the DomainBed benchmark *e.g.* [2, 105] still failed to produce noticeable performance gain when model selection (and to a lesser extent hyper-parameters selection) is done without the use of the target domain or another domain not used during training.

Several studies were further conducted to provide explanations for such failures. Hendrycks *et al.* [47] crafted several datasets exhibiting different kinds of distribution shifts and used them to benchmark many existing algorithms. Their results pointed out the high specificity of domain generalization methods. They are, often implicitly, tailored for a particular kind of distribution shift. In most cases, existing methods succeed in the case of texture shifts and fail on non-texture shifts, such as geographical shifts. Their findings are corroborated by Borlino *et al.* [11]: combined with a style-transfer data augmentation strategy (see below), which enforces a strong invariance to textures, many existing methods fail to provide an additional performance gain, which indicates a mitigation of the same texture bias.

2.3.2 *Single-source domain generalization*

Because the availability of several different source domains is an unrealistic hypothesis in some situations (such as in healthcare or defense-related tasks), methods were developed to deal with a domain shift issue with only one single domain available during training. Note

that not all methods presented below were evaluated in the single-source setting: they fall into this category as the source of the expected performance gain does not come from having several different domains available at training-time. When they are benchmarked in the multi-source setting, the different source domains are simply pulled together into a single dataset.

Many single-source methods rely on a domain shift invariance hypothesis: a prior hypothesis on what kinds of features are more likely to be spurious or missing at test-time. Ultimately, it is an assumption about the kind of domain shifts the network should be robust to. The most commonly used invariance hypothesis is the texture shift hypothesis: as a lot of real-life domain shifts are primarily textures shifts, making a network invariant to textures would render it robust to many real-life domain shifts. This objective can be achieved using style transfer, a set of deep learning methods able to transfer the style of an image, *i.e.* its textures, to another image without altering its other content [37, 55]. It can be done in a simple fashion by using style transfer as data augmentation: a model is simply trained on images whose styles are altered and diversified through a style transfer strategy. Geirhos *et al.* [39] relied on an additional paintings dataset and transferred the painting styles to the training images. Lin *et al.* [75] studied the necessity of the paintings dataset: paintings dataset against style permutations between training images. Provided that the training dataset is sufficiently large, the latter solution was as effective. Jackson *et al.* [59] instead relied on a third possibility: style embeddings are sampled randomly from a suitable distribution, thus avoiding the inconvenience of an additional dataset and creating a possibly more diverse set of styles than the training one. Samples of stylized images from [59] are given in figure 2.5. Finally, Wang *et al.* [117] went further and learned the transformation styles to produce challenging images for the network being trained. Style transfer can also be used implicitly inside the main model architecture: Zhang *et al.* [128] proposed a simple exact distribution matching layer, intertwined between the original ResNet [46] residual blocks, that enabled texture permutations between training samples. In a similar fashion, Nam *et al.* [85] proposed to use, at features level, for a given image, a style randomly interpolated between its own and that of a random secondary image. They further enforced style invariance through an adversarial learning scheme. Such methods are limited to situations where it is indeed a shift of the hypothesized nature that is encountered. However, not all real-life domain shifts encountered in computer vision tasks are texture shifts.

Other works wish to learn a larger set of predictive patterns to make the network more robust should one or several training-time predic-

Figure 2.5: Illustration of images augmented through a style transfer method, from [59]. The original image is in the top left corner, and its semantic content remains unaltered for all the different styles.

tive patterns be missing at test-time. Volpi *et al.* [114], Qiao *et al.* [94], and Zhang *et al.* [129] propose to incrementally add, to the training dataset, adversarial images crafted to maximize the classification error of the network. These images no longer contain the original obvious predictive patterns and training a network on them then forces the learning of new patterns. These strategies are inspired by adversarial training methods [54, 67] that were originally designed to improve adversarial robustness in deep networks. This paradigm relies upon the following optimization procedure:

$$\min_{\theta} \sup_{p:D(p,p_0)<\rho} \mathbb{E}_{(x,y)\sim p}[\mathcal{L}(\theta, x, y)] \tag{2.3}$$

With θ being the network's weights vector, $\mathcal{L}$ the training loss function, p_0 the labeled training data distribution (and (x, y) the training image and label pairs), D a distance (or divergence) measure over the distributions space and ρ the boundary for the space of considered distributions. For D, the Kullback-Leibler divergence (or its symmetrical counterpart, the Jensen-Shannon divergence) is sometimes used, but most of the works relying on this formalism use the Wasserstein distance. There are two main challenges with this family of approaches: firstly, only p_0 is known, the effective training distribution p has to be generated, and secondly, for such high-dimensional data, the Wasserstein distance cannot be easily computed nor enables a clear understanding of what kinds of domain shifts fall into the range of a specific ρ value. To generate the samples from the distribution p_θ defined by $\sup_{p:D(p,p_0)<\rho} \mathbb{E}_{(x,y)\sim p}[\mathcal{L}(\theta, x, y)]$, most approaches rely on an adversarial paradigm: a loss gradient ascent algorithm is run on the original images to generate images that maximize the loss from the model θ. Generated images are then added to the training set, and new images are generated to be challenging for a model trained on both

the original images and the previously generated ones. An additional constraint is used to remain in the valid distance ball, defined by ρ. Most of the time, as the distance condition cannot be enforced directly, a Lagrangian relaxation is used: a distance (different kinds are used) between a generated sample and its starting image is penalized with a strength depending on the ρ parameter. These kinds of approaches are often unstable due to the alternating adversarial learning framework and impractical due to the ever-growing training set and the difficult setting of the maximum distribution distance hyper-parameter ρ [114].

Wang *et al.* [117] used a similar approach in an online fashion, without the ever-growing dataset and combined it with the aforementioned style augmentations approaches. With the same goal of learning normally ignored patterns but with a completely different approach, Huang *et al.* [56] and Shi *et al.* [106] used a dropout-based [108] strategy to prevent the network from relying only on the most predictive patterns by muting the most useful channels or mitigating the texture bias. The main challenge of dropout-based approaches is the redundancy issue: standard dropout does not prevent the learning of the same patterns several times in different filters or neurons. This issue must be overcome to effectively learn patterns that are semantically different rather than just duplicates of the normally learned ones.

2.3.3 *Benchmarks*

Many benchmarks in the domain generalization research field originate from the domain adaptation field. Office-Home, for instance, was first introduced in [113] in a domain adaptation setting. Likewise, VLCS [33] was created by merging different domain adaptation benchmarks. PACS [72] was, however, directly introduced as a domain generalization benchmark. Samples of the PACS datasets are available in figure 2.6. These benchmarks are relatively small-scale, as there are between 10k and 15k samples for the four data domains and between 5 and 65 classes. To study domain generalization in a more realistic situation, with more classes and images than before, the DomainNet [92] benchmark (originally created for domain adaptation as well) comprises of more than half a million images divided into 345 categories. So far, all mentioned benchmarks are dedicated to a classification task. To study domain generalization in the object detection or regression setting, WILDS [64] proposed several datasets with data gathered from real-life sources exhibiting commonly encountered domain shifts and corresponding to directly useful tasks: wilderness or crops monitoring cameras in different geographical locations, histopathology images from different hospitals, satellite images of developing countries over seasonal, weather and development changes, *etc*. Likewise, a subset

Figure 2.6: Samples from the PACS dataset, from [116]. The four domains are represented: sketch, cartoon, art paintings, and photos, and illustrated with the horse and dog classes.

of Terra Incognita [7] is labeled for animals detection in wilderness cameras from different locations. For semantic segmentation, only a few datasets are publicly available. The most known and used ones are SYNTHIA [99], and Cityscapes [21]. These datasets consist in synthetic (for SYNTHIA) and real-life (for Cityscapes) urban scenes dedicated to an autonomous driving task. They are usually used to study synthetic-to-real shifts.

2.4 TEST-TIME ADAPTATION

Test-time adaption has emerged as a promising paradigm to deal with domain shifts. A network is trained normally (using stochastic gradient descent or a derivative and data augmentation), and then adapted to the data distribution encountered at use-time with a quick unsupervised procedure, as the labels are unavailable. By waiting for test-time to gather information about the target domain, in the shape of an unlabeled batch of samples from the same data distribution or even a single sample, this paradigm alleviates the main drawbacks of training-time domain generalization methods: the lack of information about the target domain, and the necessity to simultaneously adapt to all possible shifts.

The simplest test-time adaptation strategy consists of replacing the training-time statistics in the batch normalization layers with the running test batch statistics. This is now used in almost all methods [9, 53, 83, 100, 123]. This strategy was originally designed to deal with test-time images corruptions (*e.g.* blur, additive noise, *etc.*) but proved to be efficient dealing with more general domain shifts [121, 123]. A correct test statistics approximation can be reached only if all samples encountered at test-time come from the same data distribution. This is a realistic scenario for applications like autonomous driving, in which the data distribution is not expected to change over the course of a few consecutive images, but less so for networks made available online as an API where samples from a completely different distribution can be sent at the same time. In a situation where samples

of a test batch cannot be assumed to come from the same distribution, workarounds requiring only a single sample were developed by mixing test-time and training-time statistics [53, 100, 121, 123], or by using data augmentations [53]. Some solutions, such as the work of Yang *et al.* [121] or Wang *et al.* [115] further rely on test-time conditional entropy minimization, which forces highly confident predictions and prevent inconsistent features from being taken into account. To enforce this objective, only the affine parameters of the batch-normalization layers are updated to preserve the original features in case of encounters with several different domains, and to take fewer amount of time. Finally, Zhang *et al.* [126] quickly adapt the whole network to make consistent predictions between different augmentations of the same test sample. For all methods requiring a backward pass, a single gradient descent step followed by a final forward pass on the adapted model is often enough to reach a noticeable improvement. If the samples encountered at test-time can be assumed to come from the same distribution (the online adaptation scenario), the adapted model can be used and adapted again for the subsequent batches of data. If the batches of data come from different distributions (the offline adaptation scenario), the model's parameters are reset between batches to their final training values. All these strategies rely on a model trained with the standard training procedure and, as such, suffer from the shortcut-learning drawback. This limitation probably hampers these strategies' adaptation capabilities and can only be corrected during training.

2.5 CONCLUSION

Domain generalization and biased datasets issues are mitigated through a wide range of methods. However, they mostly remain an open problem, as the existing approaches' generalizability is often low, and they often rely on unrealistic hypotheses. We therefore advocate for more studies of simple and controlled situations due to the many challenges of domain generalization and the lack of significant performance gains displayed by several algorithms when the experiments are conducted rigorously. Furthermore, as we want our work to be usable in the widest range of situations, we are not focusing on invariance hypotheses, but rather on general diversity-seeking approaches. Based on these conclusions, mitigating the shortcut-learning behavior appears to be a promising strategy to diversify the learned patterns without making any assumption about the encountered domain shift. We thus first studied it in a in a strictly controlled yet challenging setting and proposed a mitigation solution, in chapter 3. Taking into account the limitations of our first approach and those of the existing works, we proposed a novel strategy to deal with the domain shift in more realistic situations, in chapter 4.

3

OVERCOMING SHORTCUT-LEARNING

In this chapter, we focus specifically on the mitigation of the shortcut-learning behavior. We first present, in section 3.1, an unlikely encounter with the shortcut-learning behavior, which is not related to a test-time domain shift, and that pushed us to research this topic. In section 3.2, we then introduce three synthetic datasets crafted to precisely measure the ability of a deep learning approach to learn patterns beyond the shortcuts. Finally, we propose two approaches to deal with the shortcut-learning flaw on our datasets. The first one, a simple additional training loss detailed in section 3.3, introduced ideas that would be reused in the next chapter but failed to generalize to all of our benchmarks. The second one, a more complex generative architecture detailed in section 3.4, yields state-of-the-art results on our benchmarks and on the realistic, heavily biased BAR dataset. We conclude this chapter in section 3.5.

3.1 FIRST APPEARANCE OF THE SHORTCUT-LEARNING BEHAVIOR

Vehicles detection from aerial images is a task of interest for Thales LAS France. However, due to the limited availability of such data and their high annotation cost, it is difficult to implement a successful deep learning approach to solve this task. One of our early work was dedicated to the addition of objects in aerial images, as a data augmentation mean. Its goal was to counterbalance the lack of instances of certain vehicle classes and avoid potentially spurious correlations between objects and backgrounds, such as boats being on blue or green waters, planes on grey landing areas, *etc*. Our experiments focused on the only publicly available large-scale aerial images dataset: the DOTA dataset [119], a collection of satellite images in the visible spectrum annotated for objects detection. Samples of this dataset are available in figure 3.1. The iSAID dataset [118] is its semantic segmentation counterpart: the exact same images are used, only labeled for a different task. The different vehicle classes (boat, car, truck, plane, helicopter) are very heavily imbalanced: while there are only a thousand helicopters, the truck and cars categories comprise of around hundred of thousands samples. This imbalance, combined with the aforementioned spurious background biases, makes the DOTA (and iSAID) dataset a challenging benchmark, even though there is no test-time domain shift.

Our approach worked as follows: objects are precisely cropped out from their original image, using their iSAID semantic segmentation

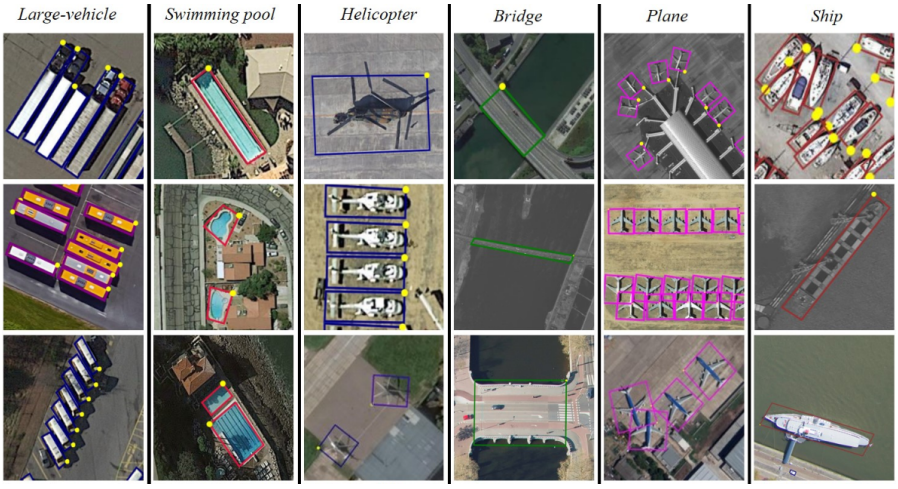

Figure 3.1: Samples from the DOTA dataset, for a subset of available classes, with their oriented bounding boxes.

label, and pasted into another image in a randomly chosen area where no object is already present. Samples of pasted vehicles are shown in figure 3.2. This data augmentation strategy, jointly used with the YOLOv3 [96, 97] deep architecture, did not enable better detection of the rare classes, despite the trials of many different parameters (the number of pasted objects per image, the relative proportion of the different classes pasted, *etc*). Sometimes, it could even perform worse than a simple training without our custom data augmentation strategy. This behavior has already been observed several times in object pasting strategies and finds its roots in the shortcut-learning phenomenon. The pasted objects exhibit boundary artifacts, due to the random non-matching area, that are picked out by the network as an effective clue to find the localization of an object. Instead of effectively learning the causal patterns, the network relies instead on visual artifacts because they are simpler and still very efficient.

All the previous works prevent the learning of such unwanted patterns using distractors, or decoys. Dwibedi *et al.* [31] and Dvornik *et al.* [30] first tried to mitigate this phenomenon by blending the boundaries, with reasonable success. The exact same scenes are generated with various blending algorithms (gaussian, Poisson, *etc.*) to prevent the learning of the blending cues rather than the causal patterns. Despite these precautions, [31] and [110] find it further necessary to add decoys, that are either objects pasted into the images but outside the scope of the task [31] (not meant to be detected and classified by the network, and as such not added to the list of object annotations) or background areas cut into the shape of an existing object and pasted onto an image [110]. Both strategies aim to add boundary artifacts in a situation where they are not predictive of sought-after object.

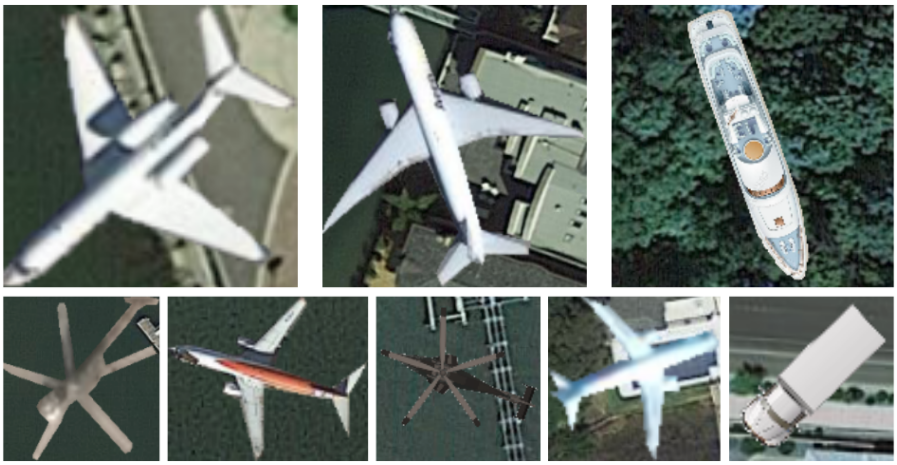

Figure 3.2: Close-up of vehicles randomly pasted into images.

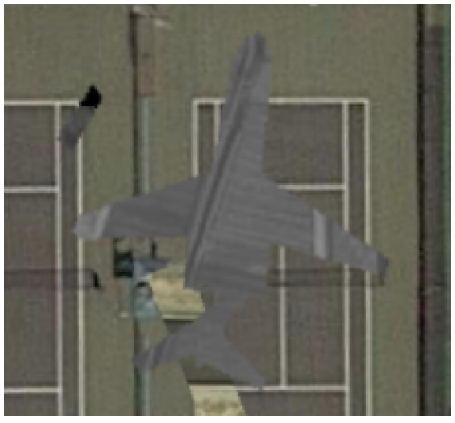

Figure 3.3: Background cut out in the shape of a plane, and pasted in an image as a decoy, to desensitize the networks to visual pasting artifacts.

This shortcut-learning situation is a very specific one that can be mitigated using explicit programming. In our case, the addition of such distractors (an example can be seen in figure 3.3) enabled us to recover the original detection performance (without any pasting data augmentation) but not to reach significantly better results. Dvornik *et al.* [30] pointed out the responsibility of the random placement in the lack of improvements and proposed context-consistent compositing of objects and images, at the cost of an increased vulnerability to spurious backgrounds. Furthermore, hard-mining appears to be mandatory for such data augmentation strategies to perform better than standard training: Liu *et al.* [77] and Mounsaveng *et al.* [82] proposed to learn the position, the orientation and a possible distortion of the pasted objects to maximize the detection errors. In [82], the method transforms the pasted objects using an auto-encoder, and in [77], a generator network is tasked with generating the additional objects from scratch, both with the same loss maximization objective. Hard-mining has proved to be an important paradigm for many training data generation approaches: for instance, Besnier *et al.* [10] found it necessary to be able to use images generated with a conditional BigGAN [12] to train a network. The use of images transformed to be specifically hard to analyze by a network plays an important part in our subsequent works.

3.2 SYNTHETIC BENCHMARKS

Following the systematic reliance on shortcut-contrary samples, we proposed a set of benchmarks that are simple to craft and to experiment with and yet challenging for debiasing, domain generalization, or shortcuts avoidance algorithms. The three benchmarks are based on the MNIST and CIFAR-10 datasets and are publicly available on github [1]. They can be seen in figure 1.2.

3.2.1 *Colored-MNIST*

In the training set of this biased dataset, every digit is colorized with a class-dependant color. All the chosen training colors are fairly different. There are furthermore no bias-contrary, or shortcut-contrary samples (images colorized differently from the majority of the other images in their class), as in [1] or label noise (images colorized with their correct class color, but with an assigned final label different from the original one), as in [93]. The validation dataset is colorized with the same colors as the training set. To precisely confront a deep network with a situation where useful training patterns are missing at test-time, the test-set color has to produce an activation that is roughly the same for all color-specific filters in the network. This way, the network is unable to use color-related information to predict the class of a digit

1 https://github.com/liris-tduboudin/Look-Beyond-Bias

and has to make use of other class correlated patterns, *e.g.* the shapes. Because we can't directly abide by this constraint, we propose to use the average of the training colors as the test color. This will suit our needs provided that no specific training color is closer to the average than any other one and that all training colors are sufficiently different from each other. For instance, if the training color of class C is closer to the average color than the other colors, the network will wrongfully believe that the test samples are all of class C. Likewise, if two training colors are close to each others, the network won't be able to easily differentiate samples of the two classes using only the color and will be driven to learn at least partially other class related patterns, *e.g.* the shape. The colors (a triplet of value in range $[0, 1]$) are therefore chosen to be on a sphere centered in the average (6 of them) and on the vertices of a cube centered in the average (the 4 left). This way, no particular color is closer to the average than at least three others. All colors are not on the sphere to avoid colors that are too similar to one another. Other works have introduced color-biased MNIST datasets [1, 3, 93]. Ours differs from theirs, most notably by the lack of counter-examples, but also by having a single domain for training and by having an unbiased dataset for testing rather than a dataset differently biased *e.g.* with different class-color combinations. One particularity of the colorized MNIST dataset is the fact that the shape and color information are spatially mingled, which prevents a simple training with cropping or cutout [24] data augmentation to find the shape information.

3.2.2 *Colored-Patch CIFAR-10*

Inspired by the work of [22], we also designed a more complex benchmark dataset based on CIFAR-10. For each of the training image, a 5x5 pixels colored patch is added in the top left corner. The patch color is the image's class color, again with no counter-examples. In the test set, all the images have the same patch color, which is the average of the training colors, like previously. For this dataset, the bias is spatially located in a tiny part of the images instead of being global as with the MNIST dataset. The colors used are the same as those used for the MNIST-based dataset.

3.2.3 *Located-Patch CIFAR-10*

Finally, to experiment on a biased dataset for which the bias is not texture-based, we create a CIFAR-10-based benchmark where it is the position of a 5x5 pixels patch that is absolutely correlated with the label *e.g.* top left corner patch for planes, bottom right corner for horses. The color of the added patch is red, for all classes as it is the

least frequent color in the CIFAR-10 dataset. For the test dataset, the average patch is added to all the images, no matter their class.

3.3 FIRST ATTEMPT IN DISCOVERING THE "HIDDEN" PATTERNS

Our first shortcut-learning mitigation approach relied on a hypothesis: we posit that, when training for a classification task, naturally learned patterns are learned because they tend to maximize inter-class specificity while minimizing intra-class variability. These learned patterns are the most effective when in discriminating between different classes. As a result, we believe that to find the "hidden" less efficient predictive patterns, we need to look for class-discriminant patterns with a higher intra-class variability than those learned naturally. That is, patterns enabling the network to better discriminate samples of the same class. Because we do not have access to additional auxiliary labels to discriminate elements within a class, we propose to expand the class-wise internal representations of a deep network.

3.3.1 *The reverse contrastive approach*

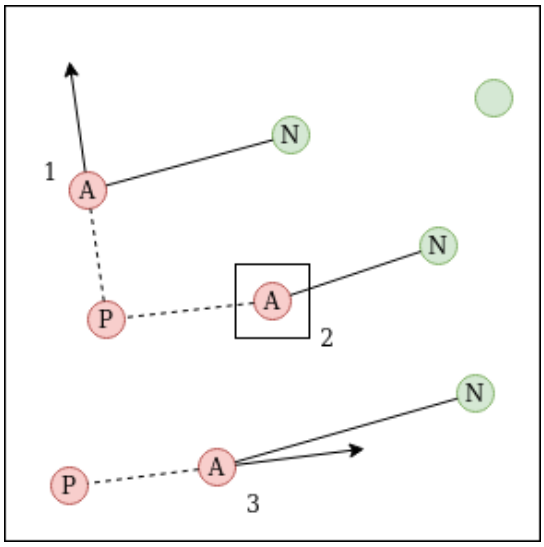

Figure 3.4: An illustration of our reverse contrastive approach on three samples in the features space. Point n°1 in the latent space is sufficiently far from its closest negative neighbor (full line) for it to be pushed away from its closest positive neighbor (dotted line). So is point n°3. Point n°2, however, is too close to a negative point to be pushed. The arrows show the moving direction of each feature.

Our method thus starts from the observation that patterns that are learned with a standard training procedure (*e.g.*, stochastic gradient descent with a cross-entropy loss for a classification task) are learned because they exhibit the maximum inter-class specificity and a fairly

low intra-class diversity. To make the network learn semantically different ways to do the task, we propose to look for class-discriminant patterns with a higher intra-class variability than those learned naturally. A higher intra-class variability means that we are able to discriminate elements inside a single class, something that is difficult with the standard training procedure (see figure 3.5, a). To find such patterns, we spread the intermediate features (the output of the convolutional features extractor F, before the last fully connected classifier layers) of elements of the same class to a certain extent. The limit in the class-wise repulsion is determined by elements of the other classes: the intra-class features are repelled from each other until we reach elements of the other classes, with a margin. This approach is akin to a partially reversed contrastive loss where positive and negative pairs are switched, but a margin must be maintained between samples of different classes. We train the network with two objectives: the conventional classification objective for both the features extractor and the classifier and the following loss for the features extractor only:

$$\begin{aligned} &min_F\{-d(f_a, f_p)\} \\ &s.t.\ d(f_a, f_p) < m \times d(f_a, f_n) \end{aligned} \tag{3.1}$$

f_a is the anchor feature map, f_p a feature map belonging to a point of the same class as f_a *i.e.* the positive sample, f_n a feature map belonging to a point of a different class than f_a, *i.e.* the negative sample, and m is the margin, a scalar to chose between 0 and 1 if we want the inter-class distance to be larger than the intra-class distance. To ease the hyper-parameter search, we use a multiplicative margin instead of an additive margin, contrary to the standard triplet loss [50]. The distance d used is the L1 distance. Feature maps are rescaled in the range $[0, 1]$ before distance calculation by using the maximum and minimum activation values computed batch-wise. This is done to avoid divergence, as there are otherwise no lower bounds for this loss. Usually, features are normalized with their L2-norm, putting them on the unit sphere, but we found better results with the min/max strategy. Practically, we use the following reverse contrastive loss (RCL):

$$\mathcal{L}_{RC} = \begin{cases} -d(f_a, f_p) \text{ if } d(f_a, f_p) < m \times d(f_a, f_n) \\ 0 \text{ otherwise} \end{cases} \tag{3.2}$$

The triplet of features used is not chosen randomly in the batch: we follow an easy-positive hard-negative sampling strategy. The anchors are sampled randomly, the positive is the closest element of the same class, the negative is also the closest element in a different class. It

is useless to push away elements of the same class that are already far from each other in the latent space: patterns extracted for these samples are already different. We compare the closest intra-class distance with the closest inter-class distance to avoid creating a mixed latent space and define precise boundaries in the latent space. The positive points used in the loss are detached from the computation graph so that only the anchor is moved in the latent space: the distance check with negative neighbors is only relevant for the anchor. This loss cannot work on its own without the classification loss: at initialization, all features are clustered in the same area of the latent space, and since the inter-class distances are small, we cannot expand the size of the intra-class clusters. The cross-entropy loss will push away the features of elements of different classes, allowing the second objective to loosen the intra-class clusters. A visualization of the proposed method can be found in figure 3.4 and the detailed algorithm in pseudo-code in algorithm 1. We found even better results when pushing away features of the same class without any kind of limitation, that is, minimizing $-d(f_a, f_p)$, or having a margin set to infinity. While counter-intuitive, this might be explained by the influence of the classification loss, which prevents a complete scattering of intra-class features and forces the differentiation of inter-class features.

Algorithm 1: Reverse Contrastive Loss

method specific hyperparameters:
- weight for the RCL α
- margin for the RCL m
while *training is not over* **do**
  sample batch of data $\{(x_i, y_i), i = 0...N\}$
  calculate cross-entropy loss on batch $\mathcal{L}_{CE}$
  calculate intermediate features $\{f_i, i = 0...N\}$
  normalize features
  for *each sample* f_i *in batch* **do**
    find closest positive $f_{p,i}$ in the batch
    find closest negative $f_{n,i}$ in the batch
    detach $f_{p,i}$ from the computation graph
    if $d(f_i, f_{p,i}) < m \times d(f_i, f_{n,i})$ **then**
      $\mathcal{L}_{RC,i} = -d(f_i, f_{p,i})$
    else
      $\mathcal{L}_{RC,i} = 0$
    end
  end
  $\mathcal{L}_{RC} = \frac{1}{N} \sum_{i=0...N} \mathcal{L}_{RC,i}$
  update model with $\mathcal{L}_{CE} + \alpha \times \mathcal{L}_{RC}$
end

3.3.2 *Baselines for comparison*

A first algorithm to find several useful patterns for classification can simply make use of dropout [108]. By zeroing activations inside the network, we naively force it to look for new patterns. A limitation of dropout is that nothing prevents the network from learning the same patterns through several filters. To avoid this redundancy phenomenon, we further implement two straightforward variants using two different regularizations: orthogonality of filters, used in [6] (more precisely, we apply the double soft orthogonality regularization) and constraint over the covariance matrix of the filters activations, used in [20]. Both regularizations are applied on the same layer as dropout, but before dropout is applied for the calculation. An orthogonality constraint for filters is supposed to prevent a filter from being close to another. The penalization of the covariance matrix of the activations is based on the idea that filters that generally activate together, or don't activate together, are probably semantically related, even though their weights might be different. These approaches constitute what we call the naive strategies.

We also implement more elaborated methods inspired by domain generalization works. First, we compare our approach with one inspired by the jigsaw puzzle multi-task strategy used in [13]. Our approach is also compared to a reconstruction multi-task strategy (inspired from [40]), where the features obtained by the features extractor are used for a classification task, with a classification head, and for an image reconstruction task, with a decoder. The feature extractor must extract sufficiently complete patterns to reconstruct the input, which is more than what is extracted with a single classification objective. The two strategies tailored precisely for the problem at hand are Representation Self Challenging [56], and Spectral Decoupling [93]. The selected methods were chosen because they improve out-of-domain generalization by finding new patterns. We did not compare ourselves with more general methods, such as the ones using style transfer, for instance, because our goal is to measure the ability of an algorithm to find less correlated patterns.

3.3.3 *Experimental setup*

Our experiments are conducted on our Colored-MNIST benchmark using a small neural network. The architecture we use was introduced in [13] to study MNIST to SVHN [88] transfer. It is composed of two convolutional layers and three fully connected layers. Max pooling operations are inserted between each convolutional. The non-linearity used is ReLU for all the layers. The convolutional layers define the feature extractor (128 channels), and the fully connected layers define the

classifier. For all experiments, we use stochastic gradient descent, with a batch size of 128, with Nesterov momentum at 0.9, a fixed learning rate of 1e-3, and an L2 weight decay at 1e-5. Models are trained for 10 epochs. There is no data augmentation. The jigsaw puzzle strategy uses an additional fully connected layer to the network, as in [13], alongside another fully connected layer used for the classification. The images are divided into 2x2 squared tiles, which are then shuffled, yielding 24 possible permutations. Each batch is used for both labels: class with the original batch sent through the network, and permutation with the shuffled batch. The decoder in the reconstruction method is composed of 4 transposed convolution layers, with a hyperbolic tangent as the last activation function. When dropout is used, the dropout rate is chosen at random between 0 and 1 for each iteration: a fixed dropout rate helps the network introduce redundancy in a simple fashion: it only has to create more redundancies than there are dropped channels. Likewise, only full channel dropout is used because of the correlation between spatially close activations in the same channel, which enable the network to recover the color all the time. For RSC, we reuse these dropout hyper-parameters, and the batch percentage (the proportion of samples per patch for which RSC is used) is fixed at 100% as we want the network to look beyond the color for every image.

Most strategies use two objectives during training: the classification cross-entropy and another objective (jigsaw puzzle, reverse contrastive loss, or reconstruction loss) or regularization (orthogonality, spectral decoupling). The weight for the classification loss is always set to 1. The weightings for the supplementary losses (or regularizations) were selected in the list $(0.001, 0.01, 0.1, 1, 5, 10)$ with the following principle: the value selected is the largest value that does not lead to a collapse of the validation accuracy. The idea is that the regularization weighting should have a positive slope for out-of-distribution accuracy, *i.e.* the larger the weight, the better the out-of-distribution accuracy up until a certain point. It is grounded in the fact that most additional objectives tend to counter the natural behavior of the network. The weight is 1.0 for the orthogonality constraint, 0.01 for the covariance constraint, 10.0 for the Jigsaw Puzzle, 1.0 for the reconstruction, 5.0 for Spectral Decoupling, and 1.0 for our reverse contrastive constraint.

During training, we select the model with the highest validation accuracy and evaluate this model on the testing data. It has been noted that domain generalization is a setting where the initialization of the network is more important than usual, and that results may vary in a greater fashion than with a training and testing dataset coming from the same distribution [66]. Therefore, we average the results over 10 runs and report the standard deviation alongside the average.

3.3.4 *Results and analysis*

Method	Validation Accuracy	Test Accuracy
Standard Training Procedure	99.8 (± 0.006)	26.0 (± 4.7)
Dropout	99.8 (± 0.01)	31.9 (± 3.1)
Dropout & Orthogonality [6]	99.8 (± 0.008)	42.7 (± 2.6)
Dropout & Covariance [20]	99.8 (± 0.002)	42.6 (± 2.0)
Jigsaw Puzzle [13]	99.8 (± 0.01)	43.0 (± 3.0)
(with early stopping @95)	98.5 (± 0.3)	59.9 (± 4.5)
Reconstruction	99.8 (± 0.007)	28.5 (± 4.8)
Spectral Decoupling [93]	99.8 (± 0.004)	47.7 (± 2.9)
(with early stopping @95)	99.4 (± 0.09)	49.8 (± 1.5)
RSC [56]	99.2 (± 0.2)	43.5 (± 5.0)
RCL-SDG (ours)		
$m = 0.2$	99.8 (± 0.007)	12.5 (± 2.6)
$m = 0.4$	99.8 (± 0.007)	19.9 (± 5.5)
$m = 0.6$	99.8 (± 0.009)	36.8 (± 5.6)
$m = 0.8$	99.7 (± 0.03)	55.7 (± 4.7)
$m = 0.9$	99.4 (± 0.08)	68.2 (± 4.0)
(with early stopping @95)	95.84 (± 0.6)	74.7 (± 8.5)
$\boldsymbol{m = \infty}$	**96.0 (± 0.3)**	**89.9 (± 0.9)**
Standard Training Procedure on the original MNIST	97.8 (± 0.12)	97.8 (± 0.12)

Table 3.1: Results of the methods on the Colored-MNIST dataset. Accuracies reported are averages over 10 runs, with the standard deviation between parenthesis. The last line indicates the accuracy reached by our backbone network on the original MNIST dataset without domain shift, it thus gives the highest reachable accuracy in the test domain for the methods.

Table 3.1 synthesizes the overall results. As can be seen, our reverse contrastive method yields a significant improvement over the previous works and the naive strategies on our dataset. Dropout compares favorably to normal training but yields far better results when used together with a regularization to prevent redundancy. This redundancy issue might explain why RSC [56] yields results only marginally above dropout and regularization. During training, we monitored the test accuracy over the epochs and noticed that jigsaw puzzle and spectral decoupling succeed to some extent but suffer from an over-fitting issue: the best test accuracy does not happen for the model with the best validation accuracy. Early stopping is useful in this situation. We employ a simple yet realistic early stopping strategy that does not use the test set: training is stopped as soon as the validation accuracy reaches a satisfying threshold, fixed here at 95% (and noted @95). This

only enables us to recover a closer accuracy than the best test accuracy but not go higher. Most methods specifically designed to prevent the network from focusing only on a subset of useful features are not effective enough to consider the problem solved on our simple dataset.

Besides the accuracy table, we also illustrate the impact of our reverse contrastive loss in figure 3.5 by looking at the latent spaces of models trained differently, through the T-SNE dimensionality reduction method [80] applied on the features. With a margin of 0.9, the class clusters are still separated, but we can see their expansions (d) compared to the standard training situation (a). Without limitation, the clusters are mixed together and can only be discriminated against one another only roughly (e). This explains why the performance in the training domain is lower: the classifier cannot perfectly separate the classes, however, the performance in the testing domain is higher. This illustrates an underlying trade-off in our method: the more the intra-class clusters are expanded, the more the network will find "hidden" useful patterns, but the more it will also learn useless patterns that are not inter-class discriminant. By comparing with the latent space of the same model trained on the original MNIST (c), we see an inefficiency of the method in finding useful patterns. The cluster sizes needed to obtain an accuracy of around 90% are larger than the size of the clusters on the original MNIST, indicating that noise and instance-specific patterns have been learned by the network. Further experiments on the Colored-CIFAR-10 and the Located-CIFAR-10 indeed showed that on more complex datasets, the RCL without limitation does not yield good results due to the image specificities being more prevalent than in MNIST. These limitations are developed in the next section.

Moreover, our approach tends to qualify the common principle in the deep learning research community that a good latent space, one able to generalize well, is supposed to have tight intra-class clusters with large margins between clusters. While this is true when there is no domain shift, it might not always hold true when so. This can be seen in figure 3.5 where a testing latent space corresponding to a large-margin tight-clusters training latent space is completely misunderstood by the network (b). On the opposite, the blurred boundaries training latent space (e) stays similar in the testing domain (f).

3.3.5 *Reverse contrastive approach summary*

We first showed that existing methods only slightly mitigate the performance drop when predictive patterns are missing at test-time, even in a simple situation like our Colored-MNIST dataset. Therefore, we proposed a counter-intuitive strategy: instead of aiming for a tight-cluster large-margin latent space, it is beneficial to try to expand the class-wise

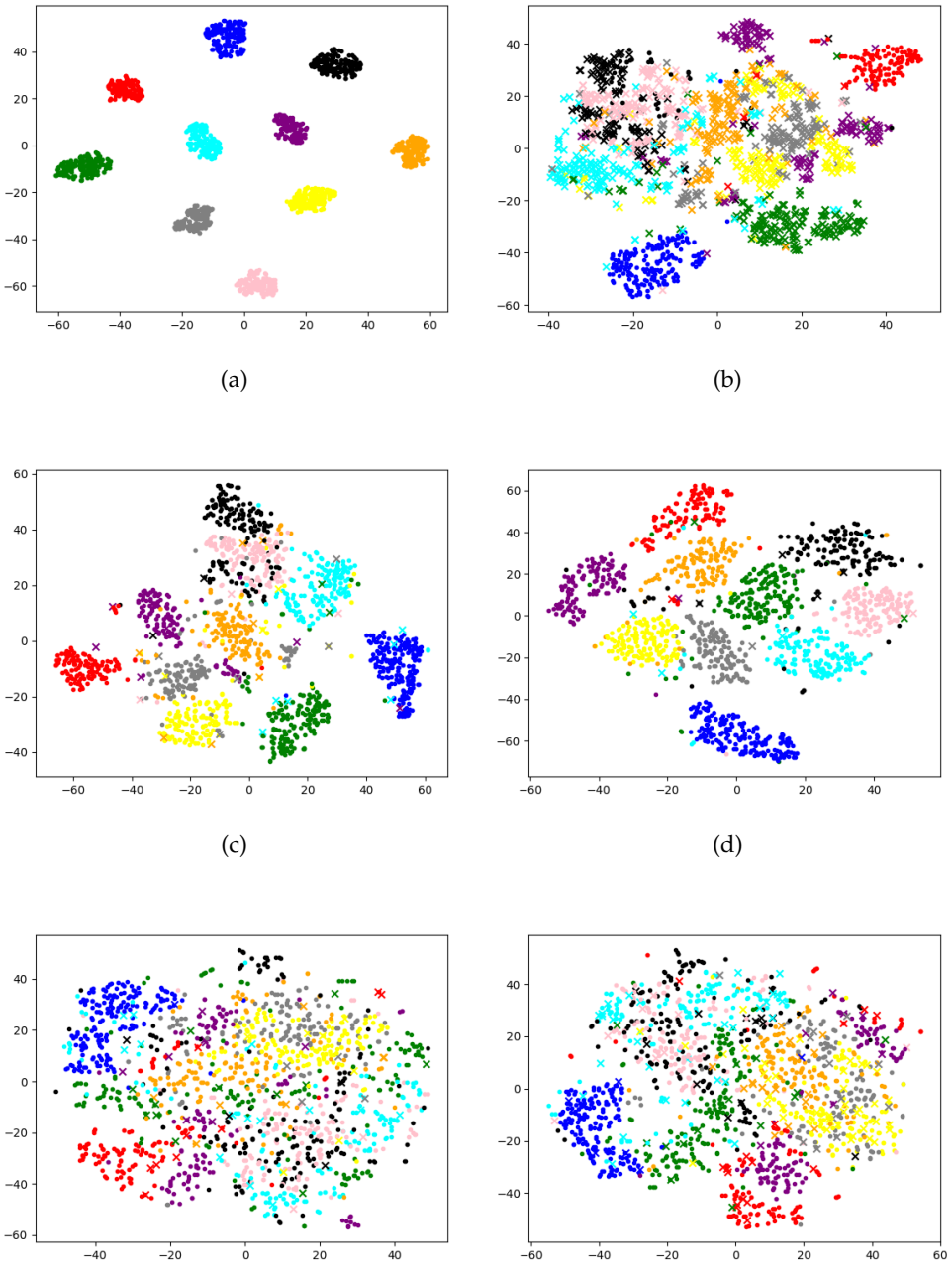

Figure 3.5: Visualisation of different latent spaces through a 2-dimensional T-SNE [80]. The two axis are the dimensions obtained by the T-SNE. *(a)* shows the validation (colored digits) latent space, for a model trained normally. *(b)* shows the test (grey digits) latent space for the same model. *(c)* shows the validation latent space for a model trained normally, on the original MNIST. *(d)* shows the validation latent space for a model trained with the RCL and a margin of 0.9. *(e)* shows the validation latent space for a model trained with the RCL without limitation. *(f)* shows the test latent space for the same model. The color of a dot is its ground truth class (not the same as the colors used for the digits) and its shape represents whether or not the network successfully predicted the correct class: circle if so, cross if not. Best viewed in color.

clusters, as the cluster size is linked to the diversity of patterns learned. Experiments have shown that our approach performs significantly better than state-of-the-art approaches on our Colored-MNIST benchmark. Note that the goal was not to compete directly against other single-source domain generalization methods on realistic benchmarks but to provide a building block for a future general algorithm.

3.4 LOOKING BEYOND BIASES WITH ENTROPIC ADVERSARIAL DATA AUGMENTATION

Despite promising results on our Colored-MNIST benchmark, the proposed reverse contrastive constraint yields results that are only on par with those obtained using the standard training procedure on both the CIFAR-10-based benchmarks and more realistic ones such as PACS (with the reverse contrastive loss adapted to a ResNet18 architecture). There are two main explanations to these failures :

- Firstly, the diversity of patterns in the training images of more complex benchmarks has a detrimental effect on the effectiveness of this reversed contrastive approach. The reverse contrastive loss pushes towards the learning of class-irrelevant patterns (patterns that are not specifically correlated with a particular class or set of classes) as it seeks to find the most different intermediate activations possible between samples of the same class. In diverse datasets, the network will easily be able to learn non-semantic patterns (such as the presence of the sky, or trees, or specific backgrounds, and colors, *etc.*). These patterns exhibit high intra-class variations and therefore enable the enlargement of features-level intra-class clusters. In the Colored-MNIST benchmark, however, the amount of non-predictive cues is not important enough for the network to be able to spread the intra-class clusters without also learning class-relevant patterns. There seem to be no simple workarounds or fixes to this flaw of the reversed contrastive loss.

- Secondly, even though a wide array of semantically different patterns is learned in the internal representation of a deep network and can be extracted at features-level, the last classification layers will rely on the simplest and most predictive features-level patterns as they are not involved in the reverse contrastive loss (and suffer as such from the shortcut-learning phenomenon). Instead of having an entire network learning only the most predictive image-level patterns, we have a few fully-connected classification layers that learn only the most predictive feature-level patterns that correspond to the most predictive image-level patterns. Observation through the application of a T-SNE of the latent space of a model trained on the Colored-CIFAR-10 benchmark (avail-

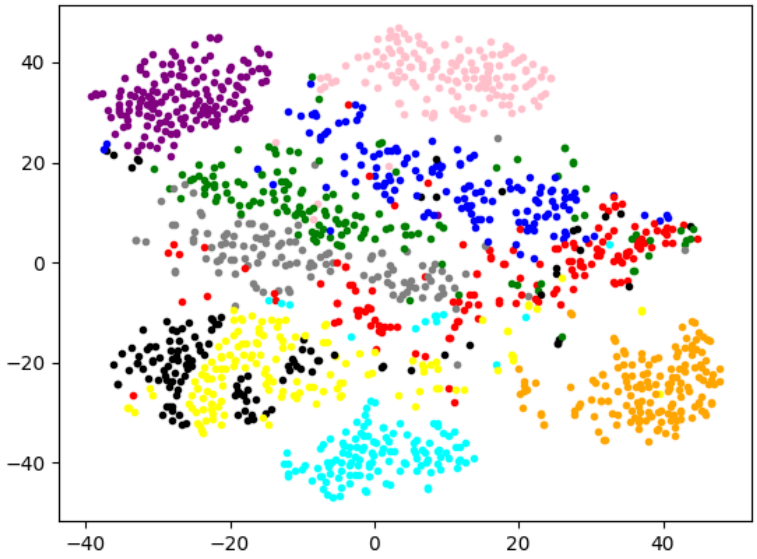

Figure 3.6: Visualisation of the validation latent space of a model trained on our Colored CIFAR-10 benchmark and with our reverse contrastive loss, through a 2-dimensional T-SNE. Despite an appearance similar to that of the Colored-MNIST latent spaces, the test accuracy is close to the original one ($\sim$ 15%).

able in figure 3.6) showed the enlarged intra-class clusters, which indicates the learning of different features than with the standard training procedure. These newly learned patterns, however, do not result in an increased accuracy at test-time, indicating that they are either non-predictive or eventually discarded by the final layers.

To deal with the shortcut-learning issue in a more general setting, we propose to instead rely on a generative approach, following a simple at first glance idea. From an image containing a prediction shortcut, we aim to create an image from which the shortcut is "removed" but for which everything else is kept intact. Better yet, we wish to create an image that no longer contains the clues currently used by a deep network to make a prediction, no matter if easy shortcuts or causal complex patterns, but still remains otherwise consistent with the original image. A classifier trained on these transformed images should learn the previously missed patterns since the most obvious and already learned ones are now missing. Since we do not assume that the shortcuts are spurious (in a real-life situation, what is naturally learned by a network will be a complex mix of causal and spurious features), we train a classifier on both the original and transformed images to learn both the shortcuts, or the naturally learned causal

patterns, and the less effective "hidden" ones. We call our approach Looking-Beyond-Bias, abbreviated as LBB.

3.4.1 *Disentangling auto-encoder with entropic data augmentations*

We implement this idea using a deep encoder-decoder architecture. We transform biased images into images that maximize the entropy of a classifier trained on both original and transformed images. With a precise control over the amount of information destroyed, via a disentanglement process, we are able to generate images where only the shortcuts are noticeably altered and replaced by patterns that are irrelevant to the prediction task.

Our method uses four distinctly trained neural networks: two classifiers (a first classifier C_0, with its convolutional features extractor F_0, and a final classifier C_f, with the same architecture), one encoder E, and one decoder D. The decoder D takes as input the output of the encoder $E(x)$, and the output of the features extractor $F_0(x)$. These two feature maps are simply resized and concatenated in the channel dimension before being sent to the decoder. We chose this fusion strategy against AdaIN-based [55] methods, such as [62], because it is simpler and makes it easier for the decoder to learn spatial information. The decoder simultaneously outputs two images (the output image has six channels, the first three being the reconstructed image D_R, and the remaining three the entropic images image D_H).

The first classifier C_0 is trained to minimize the cross-entropy on the original images without alteration to a standard training procedure. The encoder, through a latent space reconstruction loss, is made invariant to the shortcuts learned by the first classifier C_0. The decoder D_R is conditioned to output the encoder's input content with the features extractor's F_0 input shortcut. This architecture aims to produce a disentangled representation of an image between the features used by the first classifier C_0 and the remaining information $E(x)$ needed to reconstruct the original images. The remaining three channels of the decoder (D_H) are used to generate the entropic images via an adversarial training scheme. Samples of such hybrid and entropic images can be found in figure 3.8. The final classifier is simply trained to classify both the original and the entropic images coming from D_H to learn both the shortcuts and the "hidden" patterns. All the networks are simultaneously trained with their corresponding losses, although the first classifier can be trained offline beforehand and frozen during the training of the other networks. A full schema of the proposed method is available in figure 3.7.

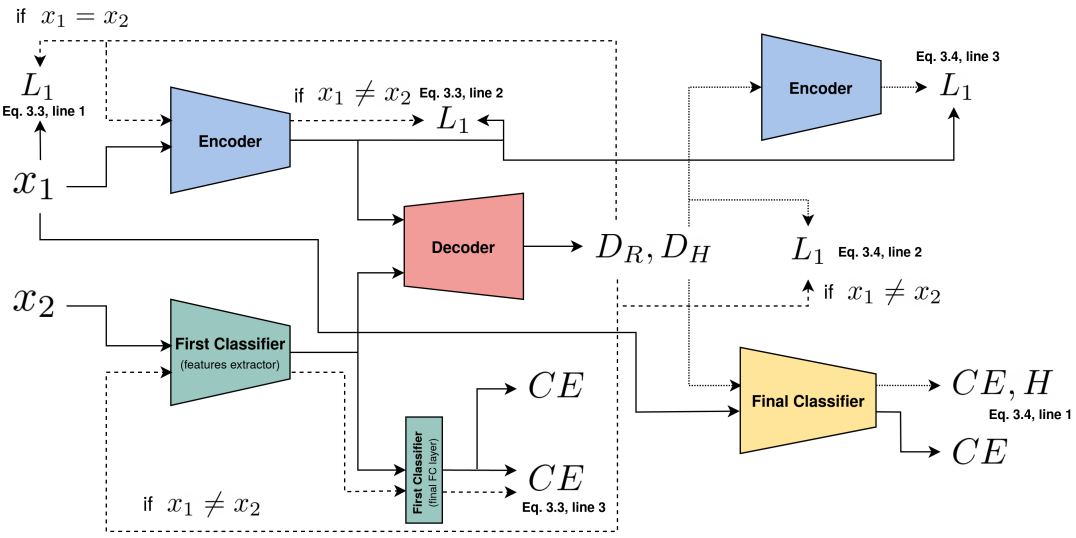

Figure 3.7: Overview of our LBB architecture. Full lines denote the algorithm pipeline for the original images, dashed lines the pipeline for the reconstructed images and the dotted lines for the entropic images. CE denotes a cross-entropy loss, H the conditional entropy of the final classifier, and L_1 an L_1 distance loss. The two encoders displayed refer to the unique one used, but were displayed twice for readability.

$$\begin{aligned}
&\mathcal{L}_R(D_R, E) = \alpha \mathbb{E}_{x\sim p_x}[||D_R(E(x), F_0(x)) - x||_1] \\
&+ \beta \mathbb{E}_{(x_1,x_2)\sim p_{x^2}}[||E(D_R(E(x_1), F_0(x_2))) - E(x_1)||_1] \\
&+ \gamma \mathbb{E}_{(x_1,x_2)\sim p_{x^2}}[-\sum_i \delta_i(C_0(x_2)) \times \\
&\log \delta_i(C_0(D_R(E(x_1), F_0(x_2))))] \qquad (3.3)
\end{aligned}$$

To properly condition the encoder and the decoder D_R to yield a disentangled representation, several training objectives are required: a reconstruction loss in the image space between the original image and the reconstructed one (Eq.3.3, line 1), an encoder latent space reconstruction loss (Eq.3.3, line 2), and a classifier prediction consistency loss (Eq.3.3, line 3). The classifier consistency loss is a cross-entropy between the prediction on an original image x_2, $C_0(x_2)$, and the prediction on a hybrid image created from $E(x_1)$ and $F_0(x_2)$: $C_0(D_R(E(x_1), F_0(x_2)))$, with δ_i being the i-th softmax coefficient: $\delta_i(y) = e^{y_i} / \sum_j e^{y_j}$. The last two losses require the simultaneous sampling of two images (x_2 can be obtained by applying a permutation on the current batch along the sample dimension). These losses enable the encoder to learn all necessary patterns for the reconstruction task with the exception of the shortcuts already provided by the features

extractor F_0, and prevent the decoder from inferring the shortcuts from the encoder representation, as the shortcut of its hybrid output $D_R(E(x_1), F_0(x_2))$ must be the one of x_2. The first and the final classifiers are not optimized with regard to these constraints.

$$\begin{aligned}\mathcal{L}_H(D_H) &= \varepsilon \mathbb{E}_{x\sim p_x}[-\sum_i q \log \delta_i(C_f(D_H(E(x), F_0(x))))] \\ &+ \mu \mathbb{E}_{x\sim p_x}[||D_H(E(x), F_0(x)) - \mathbb{E}_{f_0\sim p_{f_0}}[D_H(E(x), f_0)||_1] \\ &+ \nu \mathbb{E}_{x\sim p_x}[||E(D_H(E(x), F_0(x))) - E(x)||_1] \end{aligned} \tag{3.4}$$

The entropic output of the decoder D_H is trained to maximize the entropy of the final classifier, which is trained on both the original images and the entropic ones. The rationale behind this adversarial loss is that high-entropy images do not contain patterns that can be preferentially linked to a class and hence are more likely to be devoid of the original shortcuts. Maximizing the entropy of the first classifier only leads to changes in the bias and not to the complete removal we are aiming for. Alongside the entropy maximization (Eq. 3.4, line 1, with q the uniform probability density: $q = 1/N_c$ where N_c is the number of classes), the entropic images are subject to several constraints to avoid the destruction of all information. The first constraint lies in the decoder itself: up until the last layer, the weights are shared for both the entropic images generation and the reconstruction task. We also use the encoder latent space reconstruction loss (Eq.3.4, line 3) on the ground that the entropic images should precisely not modify what is extracted by the encoder (everything but the shortcut). Finally, we use an encoder-conditioned expected image reconstruction loss (Eq.3.4, line 2). This loss aims to drive the entropic images towards images that should already be confusing for the classifier while keeping the information not used in classification intact. It is implemented by minimizing the L_1 loss between the entropic image $D_H(E(x_1), F_0(x_1))$ and an hybrid image generated from the encoding of x_1: $D_R(E(x_1), F_0(x_2))$. Because the semantic image x_2 is randomly sampled every iteration, minimizing this loss will eventually yield the average-biased image $\mathbb{E}_{f_0\sim p_{f_0}}[D_R(E(x_1), f_0)]$.

Early experiments with a simple auto-encoder trained to maximize the entropy of a classifier and to minimize the distance to the original image showed that the balancing of the entropy maximization loss and the identity loss (L1 loss in the image space) was difficult. The adversarial auto-encoder was unstable and very often collapsed to either being the identity function or destroying all the semantic content

(details are given in the ablation study in table 3.5). To avoid these collapsing situations, a more complex architecture was needed.

3.4.2 *Baselines for comparison*

We compare ourselves with several state-of-the-art strategies from either the debiasing or the domain generalization community. A change of biases between training and testing is a domain shift, and even though domain generalization methods were not developed with such shifts in mind, it is interesting to see how they perform. We also compare ourselves with simple baselines as a reality check: first, with the standard training procedure (stochastic gradient descent with momentum, with the cross-entropy loss), then with dropout [108]. As in the reverse contrastive approach experiments, to avoid patterns redundancy issues, we experiment with the orthogonality constraint on the weights [6], and with the constraint over the norm of the covariance matrix of the intermediary activations [20]. Finally, we compare ourselves with some of the single-source domain generalization methods (Jigsaw [13], Spectral Decoupling [93], and RSC [56]) and debiasing methods (LfF [86], JTT [76] and LDD [70]) reviewed in the state-of-the-art works section.

3.4.3 *Experimental setup*

Our architecture exists in two different flavors: large-scale and small-scale. The large-scale version uses a ResNet18 [46] (adapted to the CIFAR10 and MNIST datasets as in [124]) as classifier, and a UNet [98] as encoder and decoder. The features extractor F_0 is the classifier without its last layer. The UNet is divided into a multi-output encoder and a multi-input decoder to account for the skip-connections. Each encoder's output is taken as input by the corresponding decoder's input. The semantic information from the features extractor F_0 is concatenated to the deepest encoder output before being given to the deepest decoder input. For both architectures, the optimizer used is Adam [63] with the same learning rates for all the networks. We evaluate our approach on our synthetic benchmarks and on the more realistic Biased-Action-Recognition (BAR) dataset. BAR images are divided into six activity classes and exhibit a strong (but not absolute) background bias in the training data: climbing often takes place on grey rocky background, throwing on a green baseball field, *etc*. The test set, however, is mostly made of images taken from unusual circumstances. As already stated, this dataset is a common benchmark for debiasing methods that make use of the training minority samples. No data augmentation is used for the experiments as we want to assess the effectiveness of our method without adding any predefined invariances to the training procedure. Hyper-parameters

settings and model selection are done without using the test set to mimic a situation where the test-time data distribution is unknown. For our architecture, we use the entropy loss curve to set the sensitive ε hyper-parameters: this loss should, in fact, increase continuously during training. A diminishing entropy means that the final classifier cannot keep up with the decoder's entropic images and that there is a destruction of information. The goal is to aim for the slowest increasing entropy curve possible. For the model selection with our method, we use the model at the final epoch, on the ground that training on the entropic images is much slower than training on the normal images. Training is considered complete when the entropy curve no longer increases. Finally, because the effect of the random initialization of the networks (Kaiming initialization [45], for the auto-encoder and the classification layers of the two pretrained ResNets) is greater than usual in a domain shift situation [66], results are averaged over 3 runs.

More experimental details for the large-scale experiments are given below. Because the existing baselines were never tried on our benchmarks, a hyper-parameters search was necessarily conducted beforehand.

Synthetic Datasets: the MNIST images are colorized and resized to 32x32 pixels, resulting in 3x32x32 pixel images, as the CIFAR10-based benchmarks images. For all synthetic experiments, the images are normalized with a mean and a standard deviation of [0.5,0.5,0.5]. The validation set and the test set are both made with the images from the original MNIST or CIFAR10 test sets but are biased differently. There is no pretraining of the classifiers for these datasets.

BAR: the BAR images are resized to 128x128 pixels, for computational convenience, and normalized with the following means and standard deviations: [0.485, 0.456, 0.406], and [0.229, 0.224, 0.225]. The validation set is made of 300 images that are removed from the original training set, keeping $\sim$ 1700 images for training. For all experiments on BAR, whether for our approach or existing methods, the ResNet18 is pretrained on ImageNet. Samples of the original BAR dataset can be found in figure 3.9 alongside their hybrid and entropic counterparts.

Standard Training Procedure: for all datasets, we used a learning rate of 10^{-3} with the stochastic gradient descent (SGD) with a momentum of 0.9 and trained for 100 epochs, with a batch size of 128 for all datasets (all the experiments are trained with this batch size). The test-time model is selected by best accuracy on the validation set. In the case where the best accuracy is reached several times during training (a common phenomenon since it is easy to reach 100% accuracy on

the biased datasets), the model retained is the first to reach it.

Jigsaw: images are divided into a 2x2 grid, whose patches are then shuffled. The weight for the permutation classification loss is fixed at 1.0 for BAR, and 1.0 for the synthetic datasets. Early stopping was applied as soon as the validation accuracy reached 99%. Subsequently, the model selected was the last one. Other training hyper-parameters are the same as for the standard training procedure, and this will also be the case for the next experiments unless specified otherwise.

Dropout: dropout was applied at the end of the features extractor part of the classifiers. It is before the last fully-connected layer for the ResNet18, and after the two convolutional layers for the LeNet. The zeroing probability is chosen randomly at each iteration.

Dropout & Covariance Constraint: the penalty is calculated with the L_2 norm of the covariance matrix of the features extractor activations computed over the current batch. The diagonal of the covariance matrix is fixed at 0 beforehand. The weight for the covariance penalty was fixed to 10^{-4} for all experiments.

Dropout & Orthogonality Constraint: the orthogonality constraint is calculated for all the layers, and the final constraint used is the average penalty over all layers. A particular layer's penalty is the L_2 norm of the dot product between a layer's weights and its transpose. The weight of the penalty is fixed to 1.0 for all experiments. Overall, the effects of these additional constraints compared to a simple dropout are negligible. Nonetheless, due to the apparent simplistic nature of our synthetic datasets, we deemed it necessary to try naive approaches first to ensure that a more complex one was indeed needed to reach satisfying results.

RSC: we used channel-wise dropout with a batch percentage of 100%, and the amount of channels dropped is randomly chosen at every step. Channels are sorted with respect to their usefulness for the classification on each sample in the batch, and a varying number of the most effective ones are dropped.

Spectral Decoupling: the weight for the L_2 on the raw logits of the network (before the softmax) is fixed at 0.01 for BAR, and at 1.0 for the synthetic benchmarks. Early stopping is applied when the validation accuracy reaches 99%.

LfF: the architecture was trained with Adam and a learning rate of 10^{-4}, for 100 epochs, with an amplification factor $q = 0.7$, for all the

experiments.

LDD: the architecture was trained with Adam and a learning rate of 10^{-4}, for 100 epochs. The amplification factor is the same as for LfF. The hyper-parameters specific to LDD ($\lambda_{dis}, \lambda_{swap_b}, \lambda_{swap}$) were kept to the default value: [1.0,1.0,1.0], used in the original paper for datasets similar to ours. The bias-conflicting augmentation is scheduled to be applied after the first epoch for BAR and after the default value (10k iterations) for the synthetic datasets. Without counter-examples, this parameter has no impact on the results.

JTT: one of the core principles in JTT is to train the first network only for a limited amount of epochs to avoid overfitting on the train set and keep a few numbers of misclassified train samples. For our synthetic experiments, the perfect classifier was reached before the end of the first epoch. To properly adapt the method, we stopped the training after a 99% accuracy on the current batch was reached for the tenth time since the beginning. On BAR, the first network was trained for 1 epoch before switching to training on the over-sampled dataset.

Ours (LBB): The architecture was trained for 100 epochs, with Adam and a learning rate of 10^{-4}. The hyper-parameters used were: $\alpha = 1.0$, $\beta = \gamma = 0.1$, $\varepsilon = 10^{-3}$, $\mu = 1.0$, $\nu = 0.1$, for both the experiments on the synthetic datasets and BAR.

The small-scale version uses a LeNet as classifier (as used in [13] for the MNIST-based experiments). The encoder (respectively the decoder) is a custom network with 4 convolutional (respectively transposed convolutional) layers. The architecture details are available in Table 3.2. The decoder has 256 input channels while the encoder only has 128 output channels because the LeNet features extractor (layers belonging to the features extractor are marked in bold) output also has 128 channels. There are no skip-connections, and the encoder and decoder are single-output and single-input. The hyper-parameters used in our approach for the small-scale experiments are exactly the same as in the large-scale experiments for our approach and the debiasing methods. Some domain generalization algorithms, however, required different hyper-parameters: they were only trained for 20 epochs, the weight for the logits norm minimization in Spectral Decoupling was fixed at 5.0, the weight for the permutation classification loss in Jigsaw was fixed at 10.0 and the covariance norm minimization weight used was 10^{-3}.

Table 3.2: Small-scale networks architecture.

Syntax follows PyTorch format		
LeNet	Encoder	Decoder
Conv2d(3,64,5)	Conv2d(3,32,3)	ConvTranspose2d(256,64,4,2)
ReLU	ReLU	ReLU
MaxPool2d(2,2)	Conv2d(32,64,3)	ConvTranspose2d(64,64,3)
Conv2d(64,128,5)	ReLU	ReLU
ReLU	Conv2d(64,64,3)	ConvTranspose2d(64,32,3)
MaxPool2d(2,2)	ReLU	ReLU
Linear(3200, 1024)	Conv2d(64,128,3,2)	ConvTranspose2d(32,6,3)
ReLU	ReLU	
Linear(1024,1024)		
ReLU		
Linear(1024, 10)		

3.4.4 *Results and analysis*

Our main results are available in Table 3.3 for the large-scale version. The numbers displayed are the average accuracy $\pm$ the standard deviation. Our method yields a significant accuracy improvement over the previous works on our synthetic benchmarks on the test sets while retaining a very high accuracy on the validation sets. A high accuracy reached in both validation and test indicates that both the shortcuts and the "hidden" patterns are learned: if the shortcuts are completely ignored, we expect the accuracy to be similar for the biased and unbiased datasets. The drop in validation for Colored-CIFAR-10 is most likely due to non-optimal hyper-parameters: we used the same hyper-parameters for all three synthetic datasets. Final accuracy seems to be moderately sensitive to loss weights, except for the entropy maximization weight. Other strategies perform only marginally better than the standard training procedure. It is not surprising for debiasing methods that require explicit shortcut-contrary samples. Furthermore, our architecture enables the final classifier to find all the possible patterns "hidden" behind the shortcuts: the accuracy of our method on the test datasets is roughly equal to the accuracy we get when training the same network normally on the original MNIST or CIFAR-10 datasets (without biases). On the BAR dataset, our method performs on par with the state-of-the-art debiasing methods without explicitly relying on the unusual samples. Discrepancies between original LfF results and ours are mostly due to the resizing of the images for computational convenience (128x128 in our experiments, 224x224 in the original ones). Samples of our entropic images can be found in figure 3.8 and are effectively devoid of the original shortcuts.

Small-scale results are available in Table 3.4. The small-scale architecture was not evaluated on the BAR benchmark. On the synthetic

Figure 3.8: Samples of generated images. First & second rows: original images. Third row: hybrid images with first row content and second row shortcuts. Fourth row: hybrid images with second row content and first row shortcuts. Last row: entropic images of the first row.

benchmarks, our method is still effective, but the difference between the results on the original datasets and ours is shortened compared to the large-scale version. The debiasing methods perform as badly as in the large-scale version, which was to be expected again. Interestingly, the small-scale version of the domain generalization methods performs better than their large-scale counterparts, especially on the Colored-MNIST dataset. While for the large-scale setting, all of these performed similarly on all three benchmarks, there are clear differences in the small-scale setting: most of the strategies now perform better on the Colored-MNIST benchmark. Dropout-based methods and Jigsaw yield an important increase in accuracy on Colored-MNIST, but not on the CIFAR-10-based benchmarks. Notably, Spectral Decoupling produces an increase in accuracy in all three benchmarks, even if not the best on Colored-MNIST. These results are in contradiction with the common observation that higher-capacity networks are more robust to domain shifts, on average, than smaller ones [42, 71]. It is most likely because of the very specific nature of our benchmarks domain shifts.

Table 3.3: Main results of our Look-Beyond-Bias approach.

Large-Scale Experiments (Resnet18 as classifier)								
Dataset →	Colored-MNIST		Colored-Patch CIFAR10		Located-Patch CIFAR10		BAR	
Method ↓	Val. Acc.	Test Acc.	Val. Acc.	Test Acc.	Val. Acc.	Test Acc.	Val. Acc.	Test Acc.
Standard Training Procedure	100 ± 0.0	18.8 ± 7.4	100 ± 0.0	10.3 ± 0.5	100 ± 0.0	10.0 ± 0.0	97.5 ± 0.6	49.1 ± 1.9
Dropout	100 ± 0.0	14.6 ± 6.2	100 ± 0.0	11.0 ± 1.8	100 ± 0.0	10.1 ± 0.1	98.9 ± 0.3	49.4 ± 1.0
Dropout & Orthogonality [6]	100 ± 0.0	10.2 ± 0.1	100 ± 0.0	10.0 ± 0.0	100 ± 0.0	10.0 ± 0.0	98.8 ± 0.6	48.0 ± 1.4
Dropout & Covariance [20]	93.3 ± 3.8	27.2 ± 5.4	86.4 ± 10.5	21.6 ± 2.4	67.9 ± 9.5	26.3 ± 2.2	90.6 ± 3.1	28.0 ± 3.2
Jigsaw Puzzle [13]	99.8 ± 0.3	21.5 ± 4.9	99.9 ± 0.0	17.8 ± 0.9	100 ± 0.0	12.1 ± 0.5	97.5 ± 0.3	49.8 ± 1.8
Spectral Decoupling [93]	99.9 ± 0.0	24.5 ± 2.9	100 ± 0.0	10.4 ± 0.35	100 ± 0.0	10.2 ± 0.1	96.1 ± 0.6	44.5 ± 1.8
RSC [56]	100 ± 0.0	12.5 ± 1.9	96.7 ± 5.7	10.0 ± 0.1	100 ± 0.0	10.0 ± 0.0	96.8 ± 0.9	50.2 ± 1.4
LfF [86]	100 ± 0.0	9.8 ± 1.9	100 ± 0.0	10.2 ± 0.3	100 ± 0.0	10.6 ± 0.7	97.4 ± 0.3	54.3 ± 2.3
JTT [76]	100 ± 0.0	12.5 ± 4.3	100 ± 0.0	10.4 ± 0.4	100 ± 0.0	10.0 ± 0.02	97.3 ± 0.8	50.2 ± 2.9
LDD [70]	100 ± 0.0	14.8 ± 5.3	100 ± 0.0	10.0 ± 0.2	100 ± 0.0	10.2 ± 0.3	98.3 ± 0.8	53.61 ± 2.7
Ours	$\mathbf{99.8 \pm 0.2}$	$\mathbf{97.3 \pm 0.84}$	$\mathbf{93.8 \pm 1.3}$	$\mathbf{78.9 \pm 0.34}$	$\mathbf{98.0 \pm 0.6}$	$\mathbf{75.6 \pm 3.8}$	$\mathbf{97.1 \pm 0.8}$	$\mathbf{54.4 \pm 1.1}$
Standard Training Procedure on the original datasets	99.4 ± 0.04	99.4 ± 0.04	77.4 ± 0.08	77.4 ± 0.08	77.4 ± 0.08	77.4 ± 0.08	-	-

Table 3.4: Small-scale results of our Look-Beyond-Bias approach (LeNet as classifier)

Dataset →	Colored-MNIST		Colored-Patch CIFAR10		Located-Patch CIFAR10	
Method ↓	Val. Acc.	Test Acc.	Val. Acc.	Test Acc.	Val. Acc.	Test Acc.
Standard Training Procedure	100 ± 0.0	27.2 ± 3.3	100 ± 0.0	13.2 ± 3.7	100 ± 0.0	15.5 ± 0.5
Dropout	100 ± 0.0	43.0 ± 5.0	100 ± 0.0	15.8 ± 3.5	100 ± 0.0	15.2 ± 0.8
Dropout & Orthogonality [6]	100 ± 0.0	36.5 ± 1.7	100 ± 0.0	15.6 ± 1.5	100 ± 0.0	24.0 ± 0.5
Dropout & Covariance [20]	100 ± 0.0	34.9 ± 3.9	100 ± 0.0	14.8 ± 2.0	100 ± 0.0	20.9 ± 1.8
Jigsaw Puzzle [13]	98.3 ± 0.2	65.9 ± 4.8	97.7 ± 0.3	22.0 ± 1.0	99.7 ± 0.1	21.3 ± 0.4
Spectral Decoupling [93]	99.6 ± 0.1	49.1 ± 2.5	95.4 ± 0.2	30.5 ± 1.0	96.9 ± 1.8	29.1 ± 1.2
RSC [56]	99.7 ± 0.0	45.0 ± 0.6	95.8 ± 2.0	14.5 ± 2.0	100 ± 0.0	11.4 ± 0.1
LfF [86]	100 ± 0.0	23.9 ± 5	100 ± 0.0	14.8 ± 1.8	100 ± 0.0	15.3 ± 2.1
JTT [76]	100 ± 0.0	29.2 ± 6.7	100 ± 0.0	15.4 ± 2.8	100 ± 0.0	16.1 ± 0.4
LDD [70]	100 ± 0.0	12.2 ± 2.7	100 ± 0.0	14.2 ± 4.1	100 ± 0.0	10.0 ± 0.0
Ours (LBB)	$\mathbf{99.9 \pm 0.0}$	$\mathbf{98.4 \pm 1.5}$	$\mathbf{94.3 \pm 0.2}$	$\mathbf{69.2 \pm 2.0}$	$\mathbf{97.9 \pm 0.5}$	$\mathbf{67.3 \pm 0.3}$
Standard Training Procedure on the original datasets	99.2 ± 0.04	99.2 ± 0.04	72.8 ± 0.4	72.8 ± 0.4	72.8 ± 0.4	72.8 ± 0.4

Figure 3.9: Samples of original BAR images (first 2 rows) and their generated hybrid (middle 2 rows) and entropic (last 2 rows) versions. The entropic images are almost devoid of bright colors, showing that a classifier rely heavily on them instead on the causal patterns. The blue color is not removed as it is not strongly correlated with a particular class: diving, pole vaulting and fishing images exhibit large amount of blue.

3.4.5 *Ablation study*

We also conduct an ablation study of the small-scale version of our architecture on the synthetic benchmark datasets. Our study was conducted to shed light on two questions: 1 - is the disentangling part of the architecture needed *i.e.* can't an entropy maximization and a L_1 loss between the transformed image and the original one be sufficient to yield unbiased images ? 2 - is the entropy maximization constraint needed *i.e.* can't the disentangling part with the encoder-conditioned expected image L_1 reconstruction loss be enough ? Results of the ablation study are available in Table 3.5. The reported numbers are the average accuracy on the test set of each dataset.

1 - for the first ablation experiment, the reconstruction output of the decoder D_R is no longer trained with any objectives. The entropic output D_H and the encoder E are trained with both the entropy maximization and the image reconstruction loss between the entropic image and the original one. To account for potentially different optimal hyperparameters in this situation, we conduct a study with varying entropy maximization loss weight ε, with the weight for the identity loss α fixed at 1.0. Samples of generated entropic images for this ablation are available in figure 3.11. The visualization of the samples confirms the quantitative results: too much information starts to be destroyed for $\varepsilon = 10^{-2}$, and too little for $\varepsilon = 10^{-4}$. This explains the drops in test-time accuracy at both ends of the weight range for Located-CIFAR-10, and Colored-MNIST.

2 - for the second ablation experiment, the only modification of the architecture lies in the weights used for the different losses: all the introduced losses are used for the reconstruction output D_R, but for the entropic output D_H only the encoder-conditioned image reconstruction loss remains. All other losses are removed. It is implemented by fixing all the used weights $(\alpha, \beta, \gamma, \mu)$ to 1 and all the others to 0.0 in the original architecture. Samples of generated images for this study are available in figure 3.10. The encoder-conditioned image reconstruction loss is not sufficient to ensure the complete removal of the shortcut, as can be seen with the samples of the Located-Patch CIFAR10 dataset.

Our study shows that, while either simplified architecture can yield satisfying results on a certain dataset, for a strategy to work well on all benchmarks, it has to use all the proposed constraints. Without the disentangling part (Eq. 3.3), the accuracy suffers on all datasets but especially on the Colored-MNIST dataset, no matter the ε used. We hypothesize that this is due to the spatially widespread bias in the dataset. Without the entropy maximization loss, the architecture

is not able to learn anything on the CIFAR10-based benchmarks. For the Located-Patch CIFAR10, this is due to the positional nature of the bias: without the entropy maximization constraint, the patch is simply replaced by a brown patch (similar to what can be seen in figure 3.8 with the hybrid images). A classification model trained on these samples will simply make use of a differently colored patch to make its decisions. There are no constraints to push the encoder-decoder to realistically inpaint the missing patch (such as a co-occurrence discriminator loss [91]) as the entropy maximization is effective enough. For the Colored-Patch CIFAR-10, imperfections in the disentanglement process prevent the decoder from completely removing the original color of the patch.

Table 3.5: Ablation study

Dataset →	Colored-MNIST	Colored-Patch CIFAR10	Located-Patch CIFAR10
1 - no disentanglement			
$\varepsilon = 10^{-4}$	67.1	63.5	12.1
$\varepsilon = 10^{-3}$	63.6	69.3	53.5
$\varepsilon = 10^{-2}$	31.2	62.7	60.3
$\varepsilon = 10^{-1}$	49.1	35.7	22.4
$\varepsilon = 1.0$	25.6	14.4	16.0
2 - no entropy maximization	97.7	16.0	17.3
Full Method	98.4	69.2	67.3

Figure 3.10: Samples of generated encoder-conditioned expected images for the second ablation study.

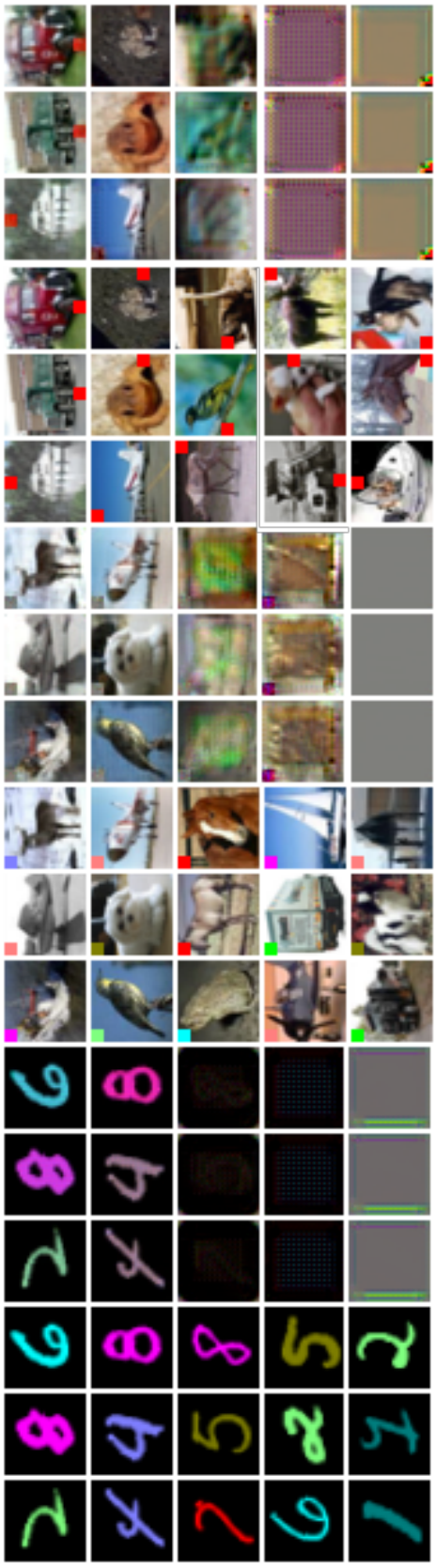

Figure 3.11: Samples of generated entropic images for the first ablation study. The first row corresponds to an entropy maximization objective weight of $\varepsilon = 10^{-4}$. This weight is increased by a factor 10 between each row. For every dataset, the first 3 columns are the original images, and the remaining 3 the corresponding entropic images.

3.4.6 *On the effect of labeling errors*

Furthermore, we want to study the behavior of existing debiasing methods when they are applied to a dataset containing annotation errors, or label noise, as their impact might be a potential drawback. To do so, we create another synthetic dataset based on MNIST. The training images are colorized as in our original Colored-MNIST dataset (every image of a particular class are colored the same), but the image label is replaced by a random one (chosen uniformly among all labels) with a certain probability p. We used $p = 0.01$, which gives 1% of randomly labeled samples. This is the order of magnitude of label noise that is encountered in usual datasets [89]. The test set is from the same data distribution, with the same label noise: there is no domain shift in this situation. We conducted this experiment with a ResNet18 as classifier for the debiasing methods and with the large-scale version of our architecture. Results of debiasing methods and of our approach on this dataset are available in Table 3.6.

Table 3.6: Debiasing and label noise

Method	Colored-MNIST w. label noise
LDD [70]	99.1
JTT [76]	99.3
LfF [86]	63.9
Standard Training	99.2
Ours (LBB)	99.0

The only method that does not succeed in dealing with the label noise is LfF [86]: training collapses after a few iterations (see figure 3.12), and does not recover. The best test accuracy is reached at the very beginning of the training before the minority samples are noticeably over-weighted, and even then, it is far below the other work's accuracy. All the other methods yield perfect results. Training debiasing methods on a dataset with wrongly labeled samples might thus have an adverse effect on the resulting accuracy, depending on the precise strategy used.

3.4.7 *On the effect of shortcut-contrary samples*

To compare our method with the debiasing strategies in a situation where they should be able to perform well, we design a version of our benchmarks with counter-examples. For 1% of the samples, the shortcut (the red patch location in the image and the patch or the digit

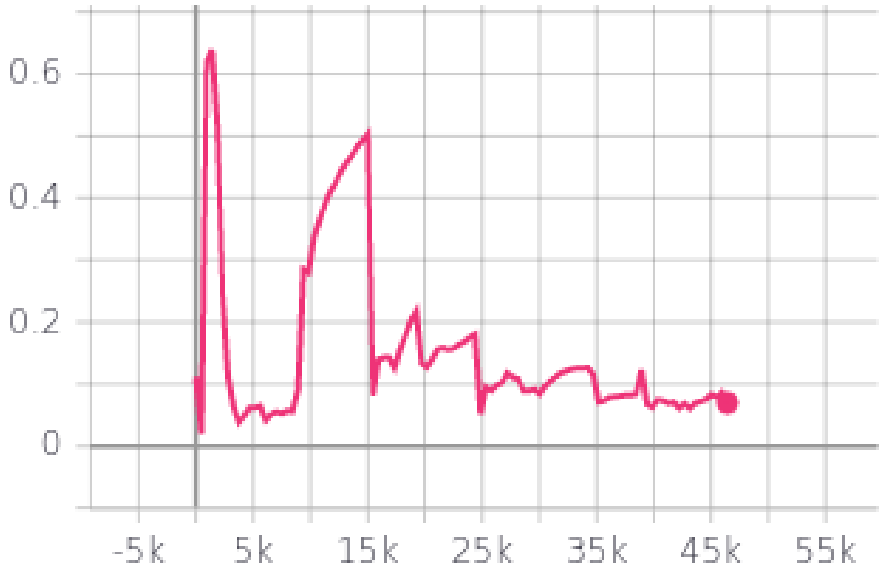

Figure 3.12: Test accuracy over training iterations for LfF, on our Colored-MNIST with label noise.

color) is chosen randomly (with equal probability for each shortcut) and not based on the image label. The test set is the same as the original one, with an average shortcut applied to each image. The small-scale results of these experiments are available in Table 3.7. The hyper-parameters used are the same as the ones used for the previous small-scale experiments, except for the starting bias-conflicting augmentation iteration in LDD [70]. In a situation with counter-examples, scheduling is important. We start the bias-conflicting augmentation after the first training epoch for all datasets.

The difference in behavior between Colored-MNIST and the other benchmarks has widened compared to the situation without counter-examples: standard training reaches satisfying accuracy on Colored-MNIST while having no positive impact on the others. Likewise, domain generalization methods only yield good results on the Colored-MNIST. As expected, the behavior of debiasing methods changes completely. They all perform almost perfectly on the MNIST-based benchmark and produce a noticeable accuracy increase on the other benchmarks. LDD's performances are, on average, on par with our method, except in the Located-CIFAR-10 dataset, where it reaches better test accuracy at the cost of a large drop in validation accuracy. Our small-scale experiments show the necessity of trying the various methods on diverse benchmarks, as the strength of the correlation between the shortcuts and the labels is not the only factor of success for a method.

Table 3.7: Small-scale results on datasets with shortcut-contrary samples

Dataset →	Colored-MNIST		Colored-Patch CIFAR10		Located-Patch CIFAR10	
Method ↓	Val. Acc.	Test Acc.	Val. Acc.	Test Acc.	Val. Acc.	Test Acc.
Standard Training Procedure	99.4 ± 0.1	65.9 ± 4.6	99.1 ± 0.0	18.1 ± 3.8	99.2 ± 0.0	27.1 ± 2.1
Dropout	99.4 ± 0.0	67.3 ± 6.1	99.2 ± 0.0	17.3 ± 4.9	99.3 ± 0.0	22.9 ± 2.9
Dropout & Orthogonality [6]	99.4 ± 0.0	68.0 ± 2.0	99.0 ± 0.1	26.8 ± 2.0	99.1 ± 0.1	27.5 ± 0.2
Dropout & Covariance [20]	99.4 ± 0.0	59.6 ± 5.0	99.14 ± 0.0	22.9 ± 1.6	99.2 ± 0.0	17.0 ± 5.4
Jigsaw Puzzle [13]	98.0 ± 0.2	72.9 ± 0.9	95.8 ± 0.5	23.6 ± 0.6	98.5 ± 0.1	20.8 ± 1.8
Spectral Decoupling [93]	98.8 ± 0.2	49.6 ± 1.3	95.3 ± 0.2	30.4 ± 1.1	96.9 ± 1.1	29.0 ± 2.4
RSC [56]	98.6 ± 0.0	72.8 ± 5.5	96.3 ± 0.4	17.7 ± 2.3	99.2 ± 0.0	14.9 ± 0.8
LfF [86]	98.7 ± 0.4	94.8 ± 0.8	89.8 ± 0.8	37.73 ± 0.2	87.3 ± 3.1	40.4 ± 1.3
JTT [76]	99.8 ± 0.0	94.1 ± 0.1	98.6 ± 0.0	30.5 ± 0.7	98.7 ± 0.0	40.7 ± 0.5
LDD [70]	99.5 ± 0.1	98.2 ± 0.2	86.1 ± 1.0	67.6 ± 0.6	86.5 ± 1.2	69.4 ± 0.7
Ours (LBB)	$\mathbf{99.8 \pm 0.1}$	$\mathbf{97.6 \pm 0.1}$	$\mathbf{88.0 \pm 0.2}$	$\mathbf{68.9 \pm 0.2}$	$\mathbf{95.9 \pm 0.3}$	$\mathbf{64.7 \pm 0.3}$
Standard Training Procedure on the original datasets	99.2 ± 0.0	99.2 ± 0.0	72.7 ± 0.4	72.7 ± 0.4	72.7 ± 0.4	72.7 ± 0.4

3.4.8 *Disentangling auto-encoder summary*

As no existing work yields satisfying results on our benchmarks, we proposed a generative architecture able to "hide" the most apparent prediction clues in an image. It relies on entropic adversarial data transformation and on the disentanglement of an image representation between the shortcuts, or what is currently relied on by a network to make a prediction, and the remaining information, inside which it is still possible to find useful patterns. We showed that it performed as well as possible, considering the classifier architectures used, on our three benchmarks. Further experiments on the BAR dataset yielded results competitive with state-of-the-art methods. This is an indication that the explicit search for counter-examples might not be necessarily needed: the information usually overlooked by neural networks is contained even in samples that exhibit the shortcuts.

3.5 CONCLUSION

3.5.1 *Summary*

In this chapter, we first proposed a new set of challenging benchmarks to precisely study the ability of deep learning methods to mitigate the shortcut-learning phenomenon. Following the observation that no existing debiasing or domain generalization methods were able to properly learn the "hidden" patterns, we made a first attempt at solving the issue on our benchmarks. The resulting reverse contrastive approach succeeded only in discovering the shapes on the Colored-MNIST benchmark but failed on the CIFAR-10-based benchmarks. A second attempt, based on a generative auto-encoder, enabled the recovery of all the "hidden" patterns in all three synthetic benchmarks and confirmed its effectiveness on the realistic biased dataset BAR. The proposed architecture relied on transforming images so that they are less likely to contain class-predictive patterns and, as such, are no longer confidently categorized by a currently trained classifier network. It is implemented by maximizing the conditional Shannon entropy of the generated images' outputs through the classifier, alongside a precise disentanglement mechanism to control the amount of information destroyed.

3.5.2 *Contributions*

Our main contributions in this chapter are the synthetic benchmarks, which are publicly released, and the second proposed method that sets a new state-of-the-art performance on the synthetic benchmarks and the BAR dataset. Our contributions shifted from the conventional debiasing paradigm. First, in debiasing works, the encountered dataset

is usually assumed to be biased, without any given procedure to assess this hypothesis. This way, the naturally learned patterns can be safely ignored without losing causal information. We are, to the best of our knowledge, the first to advocate for the learning of both the naturally learned patterns and the usually overlooked ones. Secondly, a vast majority of existing works rely on scarce shortcut-contrary samples. Our contributions, both benchmarks and methods, are designed to study and discover the "hidden" cues without the need for these particular samples.

3.5.3 *Limits and extensions*

The presented LBB method requires important computational capabilities, as it consists of the simultaneous training of between three and four deep neural networks. This can prevent its use in an industrial context. Furthermore, it is designed to learn the naturally "hidden" patterns in a classification setting. The conditional entropy loss is designed only for classification-like network outputs, and needs to be adapted for an object detection or a regression task, for instance. In some cases, *e.g.* in a semantic segmentation situation, which is a pixel-level classification task, the adaptation is straightforward, but this is not necessarily always the case. Finally, it is designed to discover and learn normally ignored patterns, but not suited to deal with more general covariate shifts. For instance, if conflicting cues are found in the test images or if test-time patterns are completely different for a particular class (in which case, only a test-time by-default class attribution reasoning can enable recognition). Based on these limitations, we now look for an algorithm able to deal with more general domain shifts. It should still rely on mitigating the shortcut-learning behavior to be efficient in a wide range of domain shifts. It should, furthermore, have the following properties: usable in the single-source domain generalization setting, lightweightness calculation-wise, and task-agnostic. To reach our objective, we propose to rely on test-time adaptation (and test-time batch-normalization, more specifically) as a starting point, because it is in line with our constraints and suitable for general domain shifts. The base representation used for test-time adaptation is further enriched by a lightweight diversity-seeking approach. This contribution is detailed in chapter 4.

4

SINGLE-SOURCE DOMAIN GENERALIZATION

As we have seen in the state-of-the-art review chapter, many existing algorithms designed to deal with general domain shifts are often unable to provide a significant performance gain compared to the standard training procedure when the real use-time conditions are properly reproduced. Furthermore, many of them require several training domains or specialize in particular domain shifts if they do not. Likewise, even if our proposed generative architecture is able to find the "hidden" features with a single training data distribution, it is not designed to deal with general domain shifts. However, the recent test-time adaptation paradigm has emerged as a possible solution to effectively increase performance, contrary to usual domain generalization training-time methods that only offer a frozen model at test-time. In this chapter, we do not propose a new test-time adaptation algorithm, instead we propose a training-time modification of the standard training procedure that enables a better adaptation by existing test-time adaptation methods. The chapter is organized as follows: section 4.1 details the idea behind our proposed approach and its specificities. The experimental details are described in section 4.2 and the results in section 4.3. Finally, we conclude this chapter in section 4.5.

4.1 LEARNING LESS GENERALIZABLE PATTERNS (L2GP)

Test-time adaptation methods suffer from a drawback that possibly limits their adaptation capability, and that should optimally be corrected at training-time (due to the availability of many labeled images). Indeed, since adapted models are obtained using a standard training procedure, they suffer from the shortcut-learning phenomenon: only the most predictive subset of predictive patterns is learned, while the less predictive patterns are disregarded entirely [7, 38, 48, 49, 93, 104, 107]. Test-time adaptation methods are more akin to a features selection, or modulation, process rather than akin to the learning of new predictive features at test-time [126]. The combination of a training-time patterns diversity-seeking approach with a test-time adaptation method may thus lead to improved results by giving the adaptation method a larger set of semantically different features to choose from.

While the previously proposed LBB method could enable a more thorough learning, its heavy computational need prevents its widespread use in an industrial context and led us to search for a simpler solu-

tion to learn a more diverse set of features. We thus propose a new lightweight method, called L2GP, which encourages a network to learn more semantically different predictive patterns than the standard training procedure. To find such different patterns, we again propose to look for predictive patterns that are less predictive than the naturally learned ones. By definition, these patterns enable correct predictions on a subset of the training data but not on all the others and are, thus, less generalizable on the training distribution. These less generalizable patterns match the ones normally ignored because of the simplicity bias of deep networks that promotes the learning of a representation with a high generalization capability on the training distribution [35, 57]. Our approach mainly consists of an additional shortcut avoidance loss that slightly encourages memorization (or over-fitting) rather than generalization by learning batch-specific patterns *i.e.* patterns that lower the loss on the running batch but with a limited effect on the other batches of data.

Our approach requires two classification layers plugged after the same features extractor: one will be tasked with learning the patterns that are normally learned (as they are not necessarily spurious and, therefore, should not be systematically ignored), and the other the normally "hidden" ones. This lightweight modification of the standard architecture, illustrated in figure 3.7, is compatible with many networks and tasks. The primary branch, consisting in the features extractor and the primary classifier, is trained to minimize the usual cross-entropy loss (algo.2, lines 11). The secondary one is trained to minimize the cross-entropy loss (algo.2, line 12) alongside a novel shortcut avoidance loss. The complete procedure is available in algorithm 2.

If we are able to update a model in a direction that lowers the loss value on a certain batch of data, but does not produce a similar decrease on another batch of the same distribution, it means that the patterns learned are both predictive as they lower the loss and generalize poorly, *i.e.* they are less predictive. These are precisely the patterns we are looking for. Our shortcut avoidance loss follows this idea. We first compute a new set of weights for the secondary branch by applying a single cross-entropy gradient ascent step to the branch weights (algo.2, lines 13-16). The gradient is computed on the original running batch, already used for the cross-entropy losses. We, then, compare the predictions of the secondary branch with the current weights and the computed altered weights (algo.2, lines 17-18). This difference in predictions constitutes our shortcut avoidance loss.

Our approach requires the sampling of two batches of data simultaneously because the shortcut avoidance loss is computed on a batch

Algorithm 2: Learning Less Generalizable Patterns (L2GP)

Method specific hyperparameters:
- weight for the shortcut avoidance loss α
- step size used for the gradient perturbation lr_+

Networks:
- features extractor f, and its weights W (ResNet18 without its last linear layer)
- first classifier c_1 (single linear layer)
- second classifier c_2 (single linear layer)

while *training is not over* **do**
- sample 2 batches of data $\{(x_i, y_i), i = 0...N-1\}, \{(\tilde{x}_i, \tilde{y}_i), i = 0...N-1\}$
- calculate the cross-entropy loss $\mathcal{L}$ on the first batch for both branches on the original weights W:
- $\mathcal{L}(f, c_1) = \frac{1}{N}\sum_i \mathcal{L}[c_1(f(W, x_i)), y_i]$
- $\mathcal{L}(f, c_2) = \frac{1}{N}\sum_i \mathcal{L}[c_2(f(W, x_i)), y_i]$
- calculate the gradient of the cross-entropy loss $\mathcal{L}$ w.r.t W on the first batch:
- $\nabla_W \mathcal{L} = \nabla_W \frac{1}{N}\sum_i \mathcal{L}[c_2(f(W, x_i)), y_i]$
- add the perturbation to the running weight W, and track this addition in the computational graph:
- $W_+ = W + lr_+ \nabla_W \mathcal{L}$
- calculate the shortcut avoidance loss on the second batch:
- $\mathcal{L}_{sa}(f, c_2) = \frac{1}{N}\sum_i ||c_2(f(W, \tilde{x}_i)) - c_2(f(W_+, \tilde{x}_i))||_1$
- update all networks to minimize $\mathcal{L}_{total}(f, c_1, c_2) = \frac{1}{2}(\mathcal{L}(f, c_1) + \mathcal{L}(f, c_2)) + \alpha \mathcal{L}_{sa}(f, c_2)$

end

At test-time: use $c_1 \circ f$ (discard c_2) combined with test-time batch normalization

of data different from the one used to compute the applied gradient. As the features learned in the applied gradient generalize from one batch of data to the other, the altered weights' predictions are a lot less accurate than the running weights' predictions (cross-entropy gradient ascent). As a result, these predictions differ greatly. By training the secondary branch to minimize the gap between both predictions, we are pushing the weights toward an area in which the applied gradient does not change the network's secondary output. This would mean that the patterns extracted for the second batch are different from the ones learned in the applied gradient. By adding the cross-entropy loss to the training procedure, we are driving the network to learn weights that are both predictive for the running classification batch but that have a low effect on the predictions of another batch and are, hence, less predictive. Note that the running network's weights are optimized with regard to both sides of the shortcut avoidance loss. The addition of the gradient must thus be tracked in the computational graph. This is akin to the MAML [34] meta-learning framework in which the starting point of a few optimization steps is itself optimized.

During the evaluation, only the first classifier is used, and the secondary one can be discarded. Indeed, the first classifier uses every available feature at its disposal, including those learned by the secondary branch, while the secondary branch only favors less simple features. Furthermore, we use test-time batch normalization (abbreviated as TTBN). This method has been chosen because of its simplicity and its wide range of applicability. We do not use the usual exponential average training mean and standard deviation (computed during training) in the batch normalization layers. Instead, we first calculate the statistics on the running test batch and use them to update an exponential average of the test statistics, as in [9, 53, 83, 100, 123], before using this estimate to normalize the features. A correct target statistics approximation can be reached only if all samples encountered at test-time come from the same data distribution. This is a realistic scenario for applications like autonomous driving, in which the data distribution is not expected to change over the course of a few consecutive images. Several methods [53, 123] provide ways to circumvent this issue if needed.

4.2 EXPERIMENTS

4.2.1 *Baselines for comparison*

We compare our approach with the standard training procedure (expected risk minimization, ERM), with several methods designed for single-source domain generalization [85, 106, 114, 117, 128, 129], with Spectral Decoupling, a method designed to reduce the shortcut-

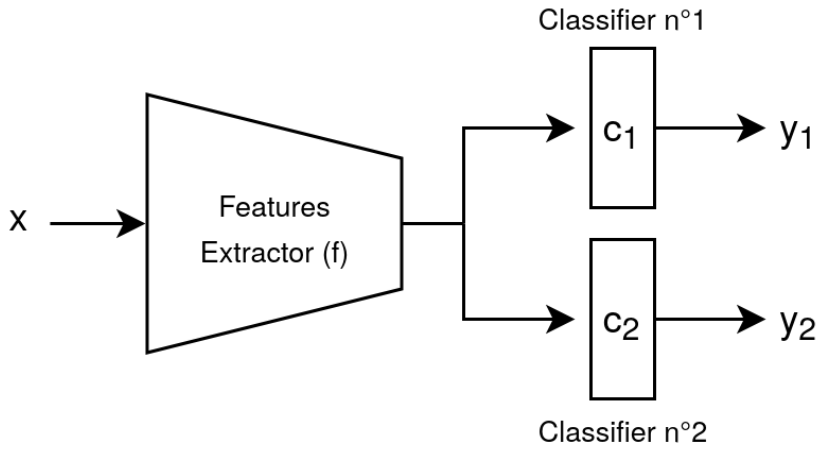

Figure 4.1: Schema of our bi-headed architecture. The naming convention is the same as the one used in algorithm 2.

learning phenomenon in deep networks [93], and with RSC, a multi-source domain generalization algorithm that does not explicitly require several training domains [56]. These baselines were selected because they yield state-of-the-art results, are representative of the main ideas in the single-source domain generalization research community, and because they have a publicly available implementation. This was a necessity as the original works' results were given without any test-time adaptation, and trained models were not provided. Our experiments are conducted on the PACS (7 classes, 4 domains, around 10k images in total), and the Office-Home (65 classes, 4 domains, around 15k images in total) benchmarks. PACS has been already sometimes used in the single-source setting, but Office-Home very rarely so in general and not in the considered baselines.

4.2.2 *Experimental setup*

For all the methods and benchmarks, we use the data augmentation described in [56] (random resized crops, color jitter, random horizontal flips, random grayscale). To avoid a target domain information leak, the models selected for the test are those with the best validation accuracy. For a particular domain used for training, 90% of the dataset is used for training and the remaining 10% for validation. The test set is obtained using another domain dataset entirely. This gives twelve training-test pairs. Experiments were conducted with a ResNet18 [46] trained for 100 epochs, with the stochastic gradient descent, a learning rate of $1e-3$, a batch size of 64, a weight decay of $1e-5$, and a Nesterov momentum of 0.9. After 80 epochs, the learning rate is divided by 10. We chose to use the same common hyper-parameters for all

baselines to precisely measure the effect of the training procedure modifications rather than the influence of a perhaps better than usual hyper-parameter. This change of hyper-parameters and differences in the model selection process (depending on the baselines, it can be best validation accuracy, last model accuracy, or best target accuracy) is responsible for some inconsistencies between the results reported in the original works and ours. In the case of SagNet [85], the differences (61.9% average accuracy on PACS in the original work, 57.9 in our own) are likely due to the model selection process. The exponential average momentum used in the batch normalization layers at test-time is set to 0.1. The gradient ascent learning rate is set to 1.0 and the α weight for the shortcut avoidance loss to 1.0 as well, for all the experiments, that is, for all the training-test pairs on both the PACS and the Office-Home datasets. These hyper-parameters were first set arbitrarily to plausible values and then confirmed to be effective on the PACS benchmark by looking at target performance. They were finally reused as is on the Office-Home benchmark. This hyper-parameters selection strategy may seem sub-optimal but is, in fact, more and more used in domain generalization problems [41, 120]: a method requiring a new and careful hyper-parameters setting for each new dataset encountered is impractical, even more so when the target data distribution is unknown and cannot thus be used to help the setting.

Comparison baselines specifics hyper-parameters are detailed below. For the experiments on the PACS datasets, on which most of the baselines were tested, we use the same hyper-parameters as in the original works. As no baselines were applied in the single-source setting on the Office-Home datasets, we used the hyper-parameters of the multi-source setting if available. If the methods did not have quantitative hyper-parameters, such as EFDM [128] with the choice of mixing-layers depths, we used the ones proposed for the PACS experiments for the ones on Office-Home. Likewise, if no rigorous hyper-parameters setting strategy was detailed in the original work, we used the PACS hyper-parameters. Finally, for the Spectral Decoupling work that was never evaluated on neither PACS nor Office-Home, we conducted a simple hyper-parameters search using a single training-test domains pair, and transferred them as is to the other pairs with the same training domain.

RSC: we used the same hyper-parameters for all experiments. The percentage of channels (or spatial cross-channel locations) to be dropped is initialized at 30% and is increased every 10 epochs linearly to reach 90% for the last ten. Spatial cross-channel locations dropout and channel all-locations dropout are applied in a mutually exclusive way with the same probability. All samples in a batch are subject to dropout.

InfoDrop: we used the same hyper-parameters for all experiments. Half the layers are subjected to the info-dropout. The dropout rate is set to 1.5, the temperature to 0.1, the bandwidth to 1.0, and the radius to 3.

ADA: for all experiments, the number of adversarial gradient ascent steps is set to 25, and the learning rate for the adversarial gradient ascent steps is set to 50. The γ and η factors are respectively set to 10.0 and 50.0. Adversarial images are added to the training set every 10 epoch.

ME-ADA: The same hyper-parameters as the ones above are used.

EFDM: For all experiments, the EFDMix layers are inserted after the first three residual blocks in the ResNet architecture.

SagNet: The randomization stage and the adversarial weight of SagNets are fixed to 3 and 0.1 for all experiments, as in the original work. A gradient clipping to 0.1 is applied to the adversarial loss.

L.t.D: α_1 and α_2 weights for the additional losses were set to 1.0, β to 0.1, for all experiments.

Spectral Decoupling: The weight of the spectral decoupling constraint (an $L2$-norm on the network's non-softmaxed output) is set to 0.001 for experiments on Office-Home Experiments, and to 0.01 for experiments on PACS.

4.3 RESULTS AND ANALYSIS

The main results of this chapter are available in Table 4.1. They were obtained as follows: for all the twelve distinct pairs of training and test domains, we calculate the average and the standard deviation of the validation and test accuracies over 3 runs (because the effect of the network's initialization on the test accuracy is greater than usual in a test-time domain shift situation). The reported numbers are then the average over all distinct pairs of the pairwise average accuracies $\pm$ the average over all distinct pairs of the pairwise standard deviation (as we are interested in the randomness of the initialization rather than the variation of accuracies between training-test pairs). Used alongside test-time batch-normalization, our method reaches a performance similar to that of EFDM [128] on the PACS datasets but exceeds it on the Office-Home datasets. When test-time batch-normalization is not used, our method remains state-of-the-art on the Office-Home dataset but falls behind the style-transfer-based methods on the PACS dataset by a noticeable margin. Besides, our approach also benefits the accuracy

	without TTBN		*with TTBN*	
Method	Avg. Val. Acc.	Avg. Test Acc.	Avg. Val. Acc.	Avg. Test Acc.
PACS dataset				
ERM	96.8 ± 0.4	52.0 ± 1.9	97.4 ± 0.3	66.1 ± 1.1
RSC [56]	97.7 ± 0.4	54.3 ± 1.8	97.2 ± 0.2	58.7 ± 1.6
InfoDrop [106]	96.6 ± 0.3	53.4 ± 2.0	95.9 ± 0.3	65.5 ± 1.0
ADA [114]	96.9 ± 0.8	55.9 ± 2.9	96.6 ± 1.1	66.5 ± 1.2
ME-ADA [129]	96.7 ± 1.3	54.7 ± 3.1	96.5 ± 0.9	66.7 ± 2.0
EFDM [128]	96.9 ± 0.5	59.6 ± 2.3	97.5 ± 0.5	$\mathbf{71.3 \pm 1.0}$
SagNet [85]	97.2 ± 0.7	57.9 ± 2.9	$\mathbf{97.8 \pm 0.7}$	62.4 ± 1.8
L.t.D [117]	97.9 ± 1.0	$\mathbf{59.9 \pm 2.7}$	97.6 ± 0.7	66.3 ± 1.5
Spectral Decoupling [93]	95.9 ± 0.4	52.9 ± 2.6	96.2 ± 0.7	66.7 ± 1.1
L2GP (ours)	$\mathbf{98.6 \pm 0.2}$	56.1 ± 2.7	96.4 ± 0.3	$\mathbf{71.3 \pm 0.6}$
Office-Home dataset				
ERM	82.0 ± 0.8	52.0 ± 0.8	81.6 ± 1.1	52.6 ± 0.6
RSC [56]	80.9 ± 0.4	49.2 ± 0.7	80.2 ± 0.5	48.9 ± 0.7
InfoDrop [106]	76.4 ± 0.8	45.9 ± 0.5	77.1 ± 0.7	46.4 ± 0.6
ADA [114]	81.2 ± 2.6	50.4 ± 0.9	80.3 ± 2.0	50.0 ± 0.7
ME-ADA [129]	78.9 ± 1.4	49.8 ± 0.6	81.4 ± 1.2	50.0 ± 0.7
EFDM [128]	82.9 ± 0.5	52.8 ± 0.6	83.3 ± 1.0	53.3 ± 0.5
SagNet [85]	81.5 ± 1.5	51.9 ± 0.7	81.1 ± 1.1	51.8 ± 0.9
L.t.D [117]	81.0 ± 1.2	50.9 ± 0.7	81.7 ± 2.7	51.2 ± 0.8
Spectral Decoupling [93]	83.8 ± 0.7	52.5 ± 0.5	82.5 ± 0.6	53.2 ± 0.3
L2GP (ours)	$\mathbf{84.0 \pm 0.6}$	$\mathbf{53.4 \pm 0.6}$	$\mathbf{83.8 \pm 0.5}$	$\mathbf{54.5 \pm 0.3}$

Table 4.1: Performances of our approach and comparison with the state-of-the-art.

on the validation sets.

We observe a completely different behavior between experiments on PACS and Office-Home. While all the existing methods improve upon the standard training procedure (ERM) on PACS, only EFDM, spectral decoupling [93], and our method yield better results on Office-Home. Likewise, while always positive, the effect of the test-time batch-normalization is much more noticeable on PACS than on Office-Home. Furthermore, it is interesting to notice that the performance gain due to the test-time batch-normalization is highly dependant on the training-time method used. Indeed, the gain is the highest when our approach or ERM is used and only reaches a result closely similar to ERM or below in most of the other cases. We hypothesize that the domain shifts of the PACS datasets are mostly textures shifts, while they are not for the Office-Home datasets. This would explain why test-time batch-normalization yields a large improvement on the PACS benchmark: the simple use of test-time statistics, that encode textures [9], is enough to significantly bridge the domain gap. It would also explain why the methods reaching the highest results [85, 117, 128] in the usual setting (without test-time batch-normalization) are all style-transfer-based methods. As our approach is not related to style transfer in any way, we are able to reach a higher accuracy on Office-Home than other existing works. Regarding the effect of different training-time methods, we hypothesize that the magnitude of the gain is related to whether the method is really learning a more diverse set of patterns or rather only weighting differently patterns that would also be learned naturally. This would explain why several methods that improve upon ERM without test-time batch normalization only perform precisely as well once it is used. Style-transfer-based methods, for instance, essentially grant a higher importance to shape-based patterns rather than texture-based patterns but not necessarily learn new patterns.

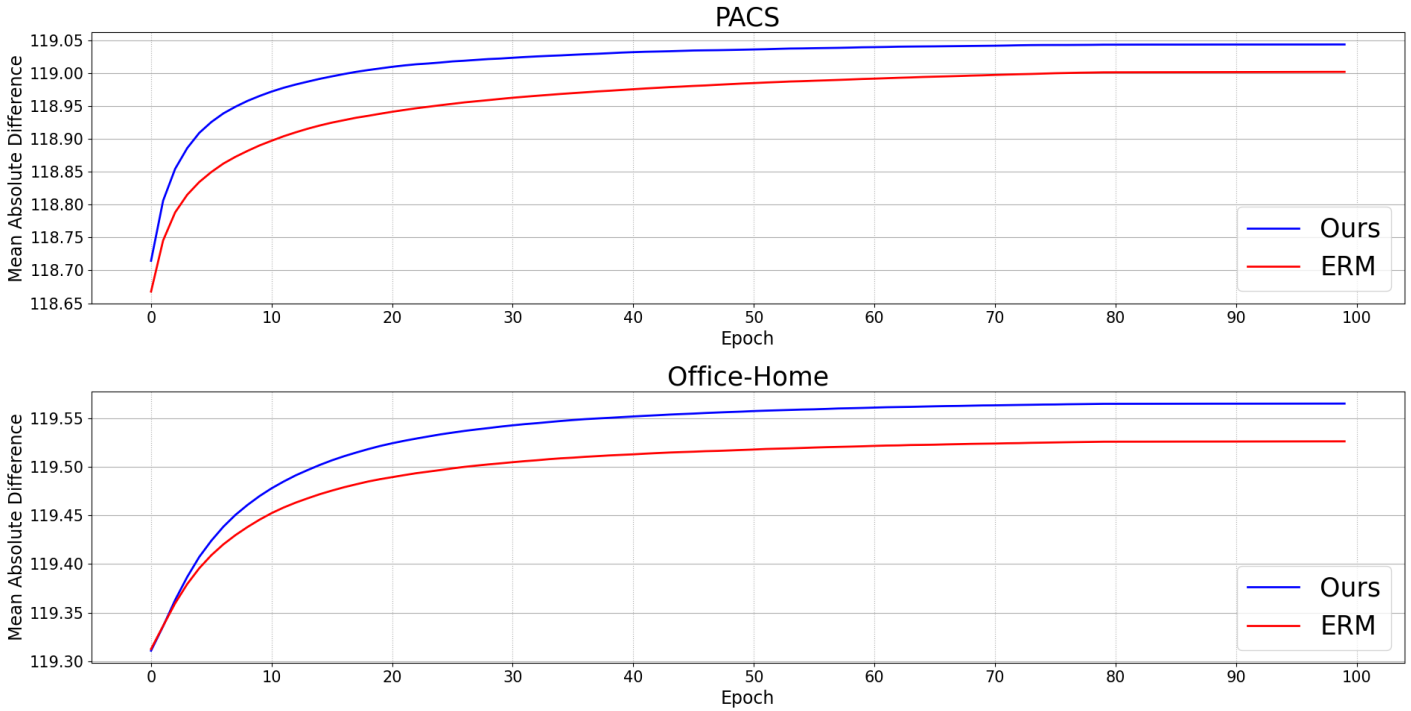

Figure 4.2: Mean absolute difference for ERM and our approach.

	without TTBN		*with TTBN*	
Ablation	Avg. Val. Acc.	Avg. Test Acc.	Avg. Val. Acc.	Avg. Test Acc.
PACS dataset				
Double branch only (A)	96.8 ± 0.6	53.4 ± 2.6	96.4 ± 0.3	67.4 ± 0.8
Detached loss term (B)	97.5 ± 0.1	52.6 ± 2.3	97.3 ± 0.3	68.2 ± 1.3
Secondary prediction branch (C)	98.0 ± 0.1	53.4 ± 2.8	96.9 ± 0.2	70.1 ± 0.4
Single branch (D)	92.8 ± 1.1	46.4 ± 4.9	93.0 ± 0.9	51.2 ± 5.1
Office-Home dataset				
Double branch only (A)	82.7 ± 0.4	52.8 ± 0.5	82.6 ± 0.3	53.5 ± 0.4
Detached loss term (B)	83.5 ± 0.7	52.7 ± 0.6	82.3 ± 0.6	54.0 ± 0.6
Secondary prediction branch (C)	81.3 ± 0.4	53.9 ± 0.7	83.8 ± 0.6	54.8 ± 0.5
Single branch (D)	82.6 ± 0.7	53.7 ± 0.4	82.0 ± 0.5	54.3 ± 0.5

Table 4.2: Ablation study

Avg. test Acc. on PACS - Avg. test Acc. on Office-Home						
$lr_+ \downarrow$ / $\alpha \rightarrow$	10^{-3}	10^{-2}	0.1	1.0	10.0	100.0
10^{-3}	66.9 - 53.7	66.8 - 53.1	67.7 - 53.2	67.0 - 53.5	67.4 - 53.2	68.5 - 53.7
10^{-2}	67.8 - 53.2	67.8 - 53.1	67.6 - 53.3	67.7 - 52.4	68.6 - 53.9	70.6 - 53.2
0.1	67.8 - 53.0	67.5 - 53.2	67.4 - 53.3	69.5 - 53.8	**71.3 - 54.7**	69.2 - 51.9
1.0	67.1 - 53.3	68.0 - 53.4	69.0 - 53.8	**71.3 - 54.4**	70.3 - 52.6	20.1 - 49.9
10.0	67.8 - 52.9	67.2 - 53.4	67.4 - 53.3	66.0 - 53.9	54.4 - 51.9	15.0 - 5.2
100.0	66.2 - 53.2	67.9 - 53.4	67.4 - 53.4	67.8 - 53.3	60.5 - 53.2	14.5 - 2.0

Table 4.3: Broad hyper-parameters sensitivity analysis.

4.4 ABLATION STUDY

We also conducted an extensive ablation study to understand and demonstrate the necessity of our choices. As a sanity check, we first study the $\alpha = 0$ situation: a single features extractor on which two classification layers are plugged in, trained only with the cross-entropy on the same batch at each iteration for both branches (line A in the table 4.2). The differences in initialization of the classifiers may have an implicit ensembling effect, as in MIMO [43], which could lead to a better out-of-distribution generalization without the need for the shortcut avoidance loss. We also study the effect of detaching from the computational graph the $c_2(f(W, \tilde{x}_i))$ term (not optimizing the features extractor with respect to this part of the loss) in the shortcut avoidance loss (line B), as this could lead to a substantial improvement in memory consumption, and as the simultaneous optimization on both terms in not needed *per se* to decrease the generalization ability of the network. Then, to show that the performance gain is effectively linked to a mitigation of the shortcut learning phenomenon, we conduct two experiments. Firstly, we study the impact of using the secondary prediction branch at test-time rather than the primary one (line C). Secondly, we study the effect of applying our shortcut avoidance loss on an architecture without the added secondary branch (line D). To further show the effect of our loss, we track during training a measure of the diversity of the learned patterns for both our approach and ERM. Inspired by [4], we use the mean absolute difference (MAD) between normalized convolutional filters f (or neurons for fully connected layers) of a certain layer, computed over all layers L of size N_L and training domains D, for an epoch t, following the equation 4.1. The results are available in figure 4.2 and show a systematic increase in the diversity of the learned patterns for our approach compared to ERM, for both benchmarks. Finally, as the tuning of hyper-parameters in the domain generalization setting is a critical issue, we conduct a broad hyper-parameters sensitivity analysis, available in table 4.3. It shows a relatively low sensitivity and a large match between hyper-parameters fit for all training-test pairs of PACS and Office-Home.

$$MAD(t) = \sum_{D} \sum_{L} \frac{1}{{N_L}^2} \sum_{i,j} ||f_{t,D,L,i} - f_{t,D,L,j}||_1 \tag{4.1}$$

The results of the ablation study outline several things: using two prediction branches without the additional loss yield a small increase of performance on both benchmarks, but it remains far below our approach, whose gain is therefore not coming from an implicit ensembling mechanism. Detaching the first half of the shortcut avoidance loss from the computational graph shows a decreased performance as well. This detachment most likely only results in a slower learning as the constraint's gradient pushes in the reverse direction of the classi-

fication loss gradient. This behavior is prevented when the features extractor is optimized with regard to both terms of the regularization: pushing in the reverse direction of the classification gradient will only slide the difference in the parameter space but not shorten the gap. Using the secondary prediction branch at test-time results in performances fairly similar to the first branch, only lower in validation. This was to be expected as the secondary branch is precisely trained so that it generalizes less on the training domain. Finally, the use of our shortcut avoidance loss applied on the original model (no added prediction branch) results in a dramatic drop in accuracy on the PACS dataset but not on the Office-Home dataset. This difference is most likely due to the higher diversity in Office-Home, that prevents the original patterns from being ignored.

Our proposed approach does not succeed on the previously proposed benchmarks based on CIFAR-10 or MNIST. The reason behind this failure lies in the specificities of these benchmarks: as the correlation between shortcuts (patches location or color, digit colors) is absolute, only a very low number of iterations (less than 100 with SGD as the optimizer, around 10 if Adam [63] is used) is needed to reach a loss value of zero. Once this step is reached, the weights of the network are no longer updated in a significant fashion as the gradient's norm is close to zero, and it points in a random direction. The time frame for our approach to learn the less generalizable patterns is thus narrow: if the gradient's norm is close to zero, the perturbed weights W_+ are very close to the original weights W. The differences between both weights predictions $\mathcal{L}_{sa}$ is thus almost non-existent and non-informative of the learned patterns and therefore have a low impact on the overall training. A possible future research direction would be to look for unified strategies able to deal with both real-life datasets and highly specific benchmarks instead of relying on combining several strategies.

4.5 CONCLUSION

4.5.1 *Summary*

In this chapter, we investigated the behavior of different single-source methods when they are used in conjunction with test-time batch-normalization, on both the PACS and Office-Home benchmarks. We showed that test-time batch-normalization always has a positive, yet highly variable, influence on the generalization ability, and that most of the time, the addition of a training-time method is superfluous. We hypothesized that this lack of additional performance was linked to the selection behavior of some algorithms, which still learn the same subset of patterns as the standard training, but weigh them differently. We thus proposed a novel approach, namely L2GP. This method

comprises of two classifiers added to a features extractor and trains the networks asymmetrically, using a data-dependant regularization, *e.g.*, shortcut avoidance loss, that slightly encourages memorization rather than generalization. By looking for predictive patterns that generalize less, it is able to learn normally "hidden" patterns. We showed that it yields state-of-the-art results on both the PACS and the Office-Home benchmarks and benefits the most to test-time batch-normalization. This approach makes use of a lightweight architecture modification in the shape of an additional final prediction layer and is not explicitly classification-specific.

4.5.2 *Contributions*

To the best of our knowledge, we are the first to investigate the conjoint use of training-time procedure and of test-time adaptation methods. Our work gave rise to an algorithm that reaches state-of-the-art performance in the challenging single-source setting on two different benchmarks. Furthermore, it is both lightweight calculation-wise and not specific to a classification task. Our approach's low count of specific hyper-parameters (weight for the shortcut avoidance loss and gradient ascent step size) and their relatively broad range of effectiveness make L2GP an interesting candidate for out-of-domain experiments on real-life datasets.

4.5.3 *Limits and extensions*

To confirm its effectiveness, further experiments on the proposed L2GP approach ought to be conducted on other benchmarks (such as the large-scale DomainNet benchmark), different network architectures (such as a ResNet-101 or a Vision Transformer), and in conjunction with more test-time adaptation methods. If the obtained results are still satisfying, a straightforward extension to object detection or semantic segmentation tasks can then be considered and experimented on.

5 CONCLUSION

This chapter concludes our work, and is organized as follows: summary of our work in section 5.1, contributions of our work in section 5.2, and future research directions and prospects in section 5.3.

5.1 SUMMARY

In this thesis, we addressed the issue of robustness to unforeseen data domain shifts. We researched a way to build a deep network model exhibiting a minimal performance drop at test-time when data coming from a distribution different than the one used for training is encountered. In our works, we focused on the mitigation of the shortcut-learning behavior of deep networks, one of the main responsible for the collapse of performance, first in controlled environments, and then to deal with general domain shifts in more realistic environments.

In Chapter 2, we review the state-of-the-art works designed to deal with different kinds of domain shifts. To deal with spurious biases that are missing at test-time, most of the existing works explicitly rely on minority samples different from their biased counterparts. More general domain shifts are partially successfully dealt with through a large diversity of training-time modifications of the standard procedure. A novel test-time adaptation paradigm is also starting to give encouraging results for robustness to general domain shifts.

In Chapter 3, we studied the shortcut-learning behavior in a controlled setting. We first describe three novel synthetic datasets that enable a precise evaluation of shortcut-learning mitigation successes or failures in a challenging setting: with an absolute correlation between shortcuts and labels, and thus no helpful minority samples. We then describe our first attempt at solving this issue on our datasets, using a reversed contrastive approach, and finally describe a successful generative approach. Our final approach relies on an auto-encoder able to "remove" from an image the cues currently used by a classification model to make its predictions. By training on both the original images and the transformed ones, the normally ignored patterns can be learned. To obtain a stable removal of predictive cues in different situations, two main components are required: first, transformed images are generated to maximize the conditional entropy of the final prediction model, and second, they are generated using only the information

currently ignored by the prediction model. This second component requires a specific disentanglement process and allows the avoidance of any involuntary leaking of already learned information into the transformed image.

The final method proposed in Chapter 3 does not generalize directly to all possible domain shifts. Furthermore, it is task-specific and requires heavy computing capabilities. These drawbacks limit its use in an industrial setting. To overcome these flaws, in Chapter 4, we proposed a new training procedure, only slightly different from the conventional ones, that prevents an over-reliance on a limited subset of patterns and instead promotes the learning of many semantically different ones. Its use alongside test-time batch-normalization, the simplest test-time adaptation method, reached state-of-the-art results on both PACS and Office-Home in the single-source setting. Our method comprises of an additional training loss that encourages the learning of batch specific patterns (*i.e.* patterns that are predictive on the current batch of data, but not on another one).

5.2 CONTRIBUTIONS

Our contributions in the field of domain generalization are the following. First, a simple set of synthetic benchmarks designed to measure the ability of a method to mitigate the shortcut-learning flaw in a difficult situation. Then, a generative auto-encoder approach able to find and learn normally "hidden" patterns even without shortcut-contrary samples. Finally, we proposed a lightweight and non classification-specific method to increase the diversity of patterns learned by a deep network, enabling its better adaptation to the encountered distribution at test-time.

Specifically, in Chapter 3, we introduced the synthetic benchmarks. Their most notable feature is the lack of shortcut-contrary samples, resulting in challenging benchmarks for which no existing methods were able to reach satisfying results despite the fact that they are based on the simple datasets MNIST and CIFAR-10.

The generative architecture, proposed in Chapter 3, is able to recover the entirety of available predictive patterns on our benchmarks. This can be inferred as our model trained on biased and transformed images reaches the same performance on the unbiased test-set as the same network architecture reaches on the original MNIST or CIFAR-10 datasets. Our approach additionally yielded results competitive with the state-of-the-art on the more realistic BAR benchmark, again without relying explicitly on the shortcut-contrary samples.

Finally, our last approach enabled to reach state-of-the-art results on both the PACS and Office-Home benchmarks in the under-explored and more realistic single-source setting. While not yet evaluated on a different task, our approach is task-agnostic (no components of the method explicitly require a particular classification loss or architecture, contrary to the entropy maximization component in the previously proposed generative architecture) and relatively lightweight. Furthermore, a low count of hyper-parameters with a low sensitivity makes it a promising candidate for real-world experiments.

5.3 FUTURE PROSPECTS

In this section, we outline some future research directions for the presented work.

PRETRAINING Discovery of hidden patterns can also be considered in the pretraining phase of the standard training procedure. Approaches interacting with the pretraining phase have very recently yielded interesting results for out-of-domain generalization. For instance, Cha *et al.* [17] tried to avoid excessive destruction of the information learned by an Imagenet-pretrained network during fine-tuning on a specific dataset. Nayman *et al.* [87] relied on both the predictiveness of the Imagenet-pretraining and the diversity of self-supervised pretrainings. Xu *et al.* [120] proposed a method able to increase style diversity that could be used during the training on ImageNet or during the fine-tuning on the smaller-scale dataset. They showed that both strategies yielded roughly the same results, and that an additional benefit comes from using the method in the two steps. Applying a diversity-seeking approach during the training of a model on ImageNet can produce weights that result in better out-of-distribution generalization (when used as pretraining for a downstream task) than weights obtained through the use of a particular method during the downstream training, due to the the higher diversity encountered in ImageNet data. Note that this strategy, while interesting to the global domain generalization research community, is not suitable for Thales's needs. Indeed, the applications developed are highly specific and do not benefit from the usual ImageNet-pretraining as they are designed for images differing greatly from the ImageNet ones. It is thus likely that extracting more diverse features from ImageNet will not have a noticeable impact on these applications.

NETWORK CAPACITY It has been noted that increasing network capacity has a positive effect on out-of-distribution generalization, both in the sense that the gap between in-domain validation and out-of-domain test performance is narrowed and that test performance is over its smaller-scale network's counterpart. Moreover, the performance

gain brought by the use of a particular method (either a modification of the training procedure, or a test-time adaptation) diminishes the larger the training model becomes [42, 71]. This positive effect on the out-of-distribution performance is not well theoretically understood, as the simplicity bias, responsible for the shortcut-learning phenomenon [104], is stronger in larger networks [35, 57] and should therefore lead to a stronger bias towards the shortcuts. Increasing the capacity of the network is not necessarily a solution, as embedded applications often require a very low computational footprint, but a possible transfer of the mechanisms behind these increased capacities to smaller-scale architecture without the additional calculation cost should be investigated.

NETWORK ARCHITECTURE The recent Transformer architecture [112] has been successfully applied to vision tasks [26, 78], in the shape of a pretrained architecture available for fine-tuning. The effectiveness of this family of architectures in the domain generalization setting has recently started to be investigated, yielding encouraging results so far [52, 125]. Transformers exhibit different intrinsic biases than convolutional architectures (most notably, they seem less biased towards textures) and, as such, should be less sensitive to certain domain shifts.

SIMPLICITY BIAS The simplicity, or inductive, bias of deep networks is responsible for their remarkable in-distribution generalization abilities. This intrinsic bias is, however, also responsible for the shortcut-learning flaw of deep networks and, as a result, partially responsible for their low ability to generalize outside of their training distribution. This observation starts to point towards a possible trade-off between in-domain and out-of-domain distribution performance, depending on the strength of the simplicity bias. Future research works should investigate this possible trade-off and explicit the relationship between shortcut-learning and simplicity bias to later propose a global paradigm that enables both in-domain and out-of-domain generalization.

BIBLIOGRAPHY

[1] Faruk Ahmed, Yoshua Bengio, Harm van Seijen, and Aaron Courville. "Systematic Generalisation with Group Invariant Predictions." In: *International Conference on Learning Representations*. 2021.

[2] Kartik Ahuja, Ethan Caballero, Dinghuai Zhang, Jean-Christophe Gagnon-Audet, Yoshua Bengio, Ioannis Mitliagkas, and Irina Rish. "Invariance Principle Meets Information Bottleneck for Out-of-Distribution Generalization." In: *Advances in Neural Information Processing Systems* (2021).

[3] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. "Invariant Risk Minimization." In: *arXiv preprint arXiv:1907.02893* (2020).

[4] Babajide O. Ayinde, Tamer Inanc, and Jacek M. Zurada. "Regularizing Deep Neural Networks by Enhancing Diversity in Feature Extraction." In: *IEEE Transactions on Neural Networks and Learning Systems* (2019).

[5] Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. "Metareg: Towards Domain Generalization Using Meta-Regularization." In: *Advances in neural information processing systems* (2018).

[6] Nitin Bansal, Xiaohan Chen, and Zhangyang Wang. "Can We Gain More from Orthogonality Regularizations in Training Deep Networks?" In: *Advances in Neural Information Processing Systems*. 2018.

[7] Sara Beery, Grant Van Horn, and Pietro Perona. "Recognition in Terra Incognita." In: *IEEE/CVF European Conference on Computer Vision*. 2018.

[8] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. "Reconciling Modern Machine Learning Practice and the Bias-Variance Trade-Off." In: *arXiv preprint arXiv:1812.11118* (2018).

[9] Philipp Benz, Chaoning Zhang, Adil Karjauv, and In So Kweon. "Revisiting Batch Normalization for Improving Corruption Robustness." In: *IEEE/CVF Winter Conference on Applications of Computer Vision*. 2021.

[10] Victor Besnier, Himalaya Jain, Andrei Bursuc, Matthieu Cord, and Patrick Pérez. "This Dataset Does not Exist: Training Models from Generated Images." In: *IEEE International Conference on Acoustics, Speech and Signal Processing*. 2020.

[11] Francesco Cappio Borlino, Antonio D'Innocente, and Tatiana Tommasi. "Rethinking Domain Generalization Baselines." In: *2020 25th International Conference on Pattern Recognition (ICPR)*. 2021.

[12] Andrew Brock, Jeff Donahue, and Karen Simonyan. "Large Scale GAN Training for High Fidelity Natural Image Synthesis." In: *International Conference on Learning Representations*. 2018.

[13] Fabio M Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. "Domain Generalization by Solving Jigsaw Puzzles." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019.

[14] Fabio Maria Carlucci, Paolo Russo, Tatiana Tommasi, and Barbara Caputo. "Hallucinating Agnostic Images to Generalize Across Domains." In: *IEEE/CVF International Conference on Computer Vision Workshop*. 2019.

[15] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. "Unsupervised Learning of Visual Features by Contrasting Cluster Assignments." In: *Advances in Neural Information Processing Systems* (2020).

[16] Junbum Cha, Hancheol Cho, Kyungjae Lee, Seunghyun Park, Yunsung Lee, and Sungrae Park. "Domain Generalization Needs Stochastic Weight Averaging for Robustness on Domain Shifts." In: *arXiv preprint arXiv:2102.08604* (2021).

[17] Junbum Cha, Kyungjae Lee, Sungrae Park, and Sanghyuk Chun. "Domain Generalization by Mutual-Information Regularization with Pre-trained Models." In: *European Conference on Computer Vision (ECCV)* (2022).

[18] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. "A Simple Framework for Contrastive Learning of Visual Representations." In: *International Conference on Machine Learning*. 2020.

[19] Xinlei Chen and Kaiming He. "Exploring Simple Siamese Representation Learning." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021.

[20] Michael Cogswell, Faruk Ahmed, Ross Girshick, Larry Zitnick, and Dhruv Batra. "Reducing Overfitting in Deep Networks by Decorrelating Representations." In: *Internation Conference on Learning Representations* (2016).

[21] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. "The Cityscapes Dataset for Semantic Urban Scene Understanding." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2016.

[22] Nikolay Dagaev, Brett D Roads, Xiaoliang Luo, Daniel N Barry, Kaustubh R Patil, and Bradley C Love. "A Too-Good-to-be-True Prior to Reduce Shortcut Reliance." In: *arXiv preprint arXiv:2102.06406* (2021).

[23] Alex J DeGrave, Joseph D Janizek, and Su-In Lee. "AI for Radiographic COVID-19 Detection Selects Shortcuts over Signal." In: *Nature Machine Intelligence* (2021).

[24] Terrance DeVries and Graham W Taylor. "Improved Regularization of Convolutional Neural Networks with Cutout." In: *arXiv preprint arXiv:1708.04552* (2017).

[25] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "Imagenet: A Large-Scale Hierarchical Image Database." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2009.

[26] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale." In: *International Conference on Learning Representations*. 2020.

[27] Thomas Duboudin, Emmanuel Dellandréa, Corentin Abgrall, Gilles Hénaff, and Liming Chen. "Encouraging Intra-Class Diversity Through a Reverse Contrastive Loss for Single-Source Domain Generalization." In: *Adversarial Robustness in the Real World Workshop of the IEEE/CVF International Conference on Computer Vision*. 2021.

[28] Thomas Duboudin, Emmanuel Dellandréa, Corentin Abgrall, Gilles Hénaff, and Liming Chen. "Look Beyond Bias with Entropic Adversarial Data Augmentation." In: *IAPR International Conference on Pattern Recognition*. 2022.

[29] Thomas Duboudin, Emmanuel Dellandréa, Corentin Abgrall, Gilles Hénaff, and Liming Chen. "Learning Less Generalizable Patterns for Better Test-Time Adaptation." In: *International Conference on Computer Vision Theory and Applications* (2023).

[30] Nikita Dvornik, Julien Mairal, and Cordelia Schmid. "On the Importance of Visual Context for Data Augmentation in Scene Understanding." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019).

[31] Debidatta Dwibedi, Ishan Misra, and Martial Hebert. "Cut, Paste and Learn: Surprisingly Easy Synthesis for Instance Detection." In: *IEEE/CVF International Conference on Computer Vision*. 2017.

[32] Simone Fabbrizzi, Symeon Papadopoulos, Eirini Ntoutsi, and Ioannis Kompatsiaris. *A Survey on Bias in Visual Datasets*. 2021.

[33] Chen Fang, Ye Xu, and Daniel N Rockmore. "Unbiased Metric Learning: On the Utilization of Multiple Datasets and Web Images for Softening Bias." In: *IEEE/CVF International Conference on Computer Vision*. 2013.

[34] Chelsea Finn, Pieter Abbeel, and Sergey Levine. "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks." In: *International Conference on Machine Learning*. 2017.

[35] Tomer Galanti and Tomaso Poggio. "SGD Noise and Implicit Low-Rank Bias in Deep Neural Networks." In: *arXiv preprint arXiv:2206.05794* (2022).

[36] Yaroslav Ganin and Victor Lempitsky. "Unsupervised Domain Adaptation by Backpropagation." In: *International Conference on Machine Learning*. 2015.

[37] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. "Image Style Transfer Using Convolutional Neural Networks." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[38] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. "Shortcut Learning in Deep Neural Networks." In: *Nature Machine Intelligence* (2020).

[39] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. "ImageNet-trained CNNs are Biased towards Texture; Increasing Shape Bias Improves Accuracy and Robustness." In: *International Conference on Learning Representations*. 2019.

[40] Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. "Domain Generalization for Object Recognition with Multi-Task Autoencoders." In: *IEEE/CVF International Conference on Computer Vision*. 2015.

[41] Tejas Gokhale, Rushil Anirudh, Jayaraman J Thiagarajan, Bhavya Kailkhura, Chitta Baral, and Yezhou Yang. "Improving Diversity with Adversarially Learned Transformations for Domain Generalization." In: *arXiv preprint arXiv:2206.07736* (2022).

[42] Ishaan Gulrajani and David Lopez-Paz. "In Search of Lost Domain Generalization." In: *International Conference on Learning Representations*. 2021.

[43] Marton Havasi, Rodolphe Jenatton, Stanislav Fort, Jeremiah Zhe Liu, Jasper Snoek, Balaji Lakshminarayanan, Andrew Mingbo Dai, and Dustin Tran. "Training Independent Subnetworks for Robust Prediction." In: *International Conference on Learning Representations*. 2021.

[44] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. "Mask R-CNN." In: *IEEE/CVF International Conference on Computer Vision*. 2017.

[45] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification." In: *IEEE/CVF International Conference on Computer Vision*. 2015.

[46] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep Residual Learning for Image Recognition." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2016.

[47] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. "The Many Faces of Robustness: A Critical Analysis of Out-of-Distribution Generalization." In: *IEEE/CVF International Conference on Computer Vision*. 2021.

[48] Katherine Hermann, Ting Chen, and Simon Kornblith. "The Origins and Prevalence of Texture Bias in Convolutional Neural Networks." In: *Advances in Neural Information Processing Systems* (2020).

[49] Katherine Hermann and Andrew Lampinen. "What Shapes Feature Representations? Exploring Datasets, Architectures, and Training." In: *Advances in Neural Information Processing Systems* (2020).

[50] Elad Hoffer and Nir Ailon. "Deep Metric Learning Using Triplet Network." In: *International Workshop on Similarity-Based Pattern Recognition*. 2015.

[51] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. "Cycada: Cycle-Consistent Adversarial Domain Adaptation." In: *International Conference on Machine Learning*. 2018.

[52] Zhi Hou, Baosheng Yu, and Dacheng Tao. "BatchFormer: Learning to Explore Sample Relationships for Robust Representation Learning." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022.

[53] Xuefeng Hu, Gokhan Uzunbas, Sirius Chen, Rui Wang, Ashish Shah, Ram Nevatia, and Ser-Nam Lim. "Mixnorm: Test-Time Adaptation Through Online Normalization Estimation." In: *arXiv preprint arXiv:2110.11478* (2021).

[54] Ruitong Huang, Bing Xu, Dale Schuurmans, and Csaba Szepesvári. "Learning with a Strong Adversary." In: *arXiv preprint arXiv:1511.03034* (2015).

[55] Xun Huang and Serge Belongie. "Arbitrary Style Transfer in Real-Time with Adaptive Instance Normalization." In: *IEEE/CVF International Conference on Computer Vision*. 2017.

[56] Zeyi Huang, Haohan Wang, Eric P. Xing, and Dong Huang. "Self-Challenging Improves Cross-Domain Generalization." In: *IEEE/CVF European Conference on Computer Vision*. 2020.

[57] Minyoung Huh, Hossein Mobahi, Richard Zhang, Brian Cheung, Pulkit Agrawal, and Phillip Isola. "The Low-Rank Simplicity Bias in Deep Networks." In: *arXiv preprint arXiv:2103.10427* (2021).

[58] Sergey Ioffe and Christian Szegedy. "Batch normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift." In: *International Conference on Machine Learning*. 2015.

[59] Philip TG Jackson, Amir Atapour Abarghouei, Stephen Bonner, Toby P Breckon, and Boguslaw Obara. "Style Augmentation: Data Augmentation via Style Randomization." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2019.

[60] Byungju Kim, Hyunwoo Kim, Kyungsu Kim, Sungjin Kim, and Junmo Kim. "Learning Not to Learn: Training Deep Neural Networks With Biased Data." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019.

[61] Donghyun Kim, Kaihong Wang, Stan Sclaroff, and Kate Saenko. "A Broad Study of Pre-training for Domain Generalization and Adaptation." In: *arXiv preprint arXiv:2203.11819* (2022).

[62] Eungyeup Kim, Jihyeon Lee, and Jaegul Choo. "Biaswap: Removing Dataset Bias with Bias-Tailored Swapping Augmentation." In: *IEEE/CVF International Conference on Computer Vision*. 2021.

[63] Diederik P Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization." In: *International Conference on Learning Representations*. 2015.

[64] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. "Wilds: A Benchmark of In-the-Wild Distribution Shifts." In: *International Conference on Machine Learning*. 2021.

[65] Alex Krizhevsky, Geoffrey Hinton, et al. "Learning Multiple Layers of Features from Tiny Images." In: (2009).

[66] David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Remi Le Priol, and Aaron Courville. "Out-of-Distribution Generalization via Risk Extrapolation (rex)." In: *arXiv preprint arXiv:2003.00688* (2020).

[67] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. "Adversarial Machine Learning at Scale." In: *International Conference on Learning Representations*. 2016.

[68] Yann LeCun, Y. Bengio, and Geoffrey Hinton. "Deep Learning." In: *Nature* 521 (May 2015), pp. 436–44.

[69] Yann LeCun, Corinna Cortes, and CJ Burges. "MNIST Dandwritten Digit Database." In: *ATT Labs. Available: http://yann.lecun.com/exdb/mnist* (2010).

[70] Jungsoo Lee, Eungyeup Kim, Juyoung Lee, Jihyeon Lee, and Jaegul Choo. "Learning Debiased Representation via Disentangled Feature Augmentation." In: *Advances in Neural Information Processing Systems* (2021).

[71] Da Li, Henry Gouk, and Timothy Hospedales. "Finding lost DG: Explaining Domain Generalization via Model Complexity." In: *arXiv preprint arXiv:2202.00563* (2022).

[72] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M. Hospedales. "Deeper, Broader and Artier Domain Generalization." In: *IEEE/CVF International Conference on Computer Vision*. 2017.

[73] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. "Learning to Generalize: Meta-learning for Domain Generalization." In: *AAAI conference on artificial intelligence*. 2018.

[74] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. "Domain Generalization with Adversarial Feature Learning." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018.

[75] Hubert Lin, Mitchell Van Zuijlen, Sylvia C Pont, Maarten WA Wijntjes, and Kavita Bala. "What Can Style Transfer and Paintings Do For Model Robustness?" In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021.

[76] Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. "Just Train Twice: Improving Group Robustness without Training Group Information." In: *International Conference on Machine Learning*. 2021.

[77] Lanlan Liu, Michael Muelly, Jia Deng, Tomas Pfister, and Li-Jia Li. "Generative Modeling for Small-Data Object Detection." In: *IEEE/CVF International Conference on Computer Vision*. 2019.

[78] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. "Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows." In: *IEEE/CVF International Conference on Computer Vision*. 2021.

[79] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. "Deep Learning Face Attributes in the Wild." In: *IEEE/CVF International Conference on Computer Vision*. 2015.

[80] Laurens van der Maaten and Geoffrey Hinton. "Visualizing Data Using t-SNE." In: *Journal of Machine Learning Research* (2008).

[81] Lucas Mansilla, Rodrigo Echeveste, Diego H Milone, and Enzo Ferrante. "Domain Generalization via Gradient Surgery." In: *IEEE/CVF International Conference on Computer Vision*. 2021.

[82] Saypraseuth Mounsaveng, David Vazquez, Ismail Ben Ayed, and Marco Pedersoli. "Adversarial Learning of General Transformations for Data Augmentation." In: *International Conference on Learning Representations, LLD workshop*. 2019.

[83] Zachary Nado, Shreyas Padhy, D Sculley, Alexander D'Amour, Balaji Lakshminarayanan, and Jasper Snoek. "Evaluating Prediction-Time Batch Normalization for Robustness under Covariate Shift." In: *arXiv preprint arXiv:2006.10963* (2020).

[84] Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. "Reducing Domain Gap via Style-Agnostic Networks." In: *arXiv preprint arXiv:1910.11645* (2019).

[85] Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. "Reducing Domain Gap by Reducing Style Bias." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021.

[86] Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. "Learning from Failure: Training Debiased Classifier from Biased Classifier." In: *Advances in Neural Information Processing Systems*. 2020.

[87] Niv Nayman, Avram Golbert, Asaf Noy, Tan Ping, and Lihi Zelnik-Manor. "Diverse Imagenet Models Transfer Better." In: *arXiv preprint arXiv:2204.09134* (2022).

[88] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. "Reading Digits in Natural Images with Unsupervised Feature Learning." In: *Advances in Neural Information Processing Systems Workshop*. 2011.

[89] Curtis G Northcutt, Anish Athalye, and Jonas Mueller. "Pervasive Label Errors in Test Sets Destabilize Machine Learning Benchmarks." In: *Advances in Neural Information Processing Systems*. 2021.

[90] Giambattista Parascandolo, Alexander Neitz, Antonio Orvieto, Luigi Gresele, and Bernhard Schölkopf. "Learning Explanations that are Hard to Vary." In: *International Conference on Learning Representations*. 2021.

[91] Taesung Park, Jun-Yan Zhu, Oliver Wang, Jingwan Lu, Eli Shechtman, Alexei Efros, and Richard Zhang. "Swapping Autoencoder for Deep Image Manipulation." In: *Advances in Neural Information Processing Systems*. 2020.

[92] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. "Moment Matching for Multi-Source Domain Adaptation." In: *IEEE/CVF International Conference on Computer Vision*. 2019.

[93] Mohammad Pezeshki, Sékou-Oumar Kaba, Yoshua Bengio, Aaron Courville, Doina Precup, and Guillaume Lajoie. "Gradient Starvation: A Learning Proclivity in Neural Networks." In: *arXiv preprint arXiv:2011.09468* (2020).

[94] Fengchun Qiao, Long Zhao, and Xi Peng. "Learning to Learn Single Domain Generalization." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020.

[95] Joaquin Quinonero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. *Dataset Shift in Machine Learning*. 2008.

[96] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. "You Only Look Once: Unified, Real-Time Object Detection." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2016.

[97] Joseph Redmon and Ali Farhadi. "Yolov3: An Incremental Improvement." In: *arXiv preprint arXiv:1804.02767* (2018).

[98] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-net: Convolutional Networks for Biomedical Image Segmentation." In: *International Conference on Medical image computing and computer-assisted intervention*. 2015.

[99] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M Lopez. "The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2016.

[100] Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. "Improving Robustness against Common Corruptions by Covariate Shift Adaptation." In: *Advances in Neural Information Processing Systems* (2020).

[101] Luca Scimeca, Seong Joon Oh, Sanghyuk Chun, Michael Poli, and Sangdoo Yun. "Which Shortcut Cues Will DNNs Choose? A Study from the Parameter-Space Perspective." In: *International Conference on Learning Representations*. 2021.

[102] Mattia Segu, Alessio Tonioni, and Federico Tombari. "Batch Normalization Embeddings for Deep Domain Generalization." In: *arXiv preprint arXiv:2011.12672* (2020).

[103] Seonguk Seo, Yumin Suh, Dongwan Kim, Jongwoo Han, and Bohyung Han. "Learning to Optimize Domain Specific Normalization with Domain Augmentation for Domain Generalization." In: *arXiv preprint arXiv:1907.04275* (2019).

[104] Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. "The Pitfalls of Simplicity Bias in Neural Networks." In: *Advances in Neural Information Processing Systems*. 2020.

[105] Soroosh Shahtalebi, Jean-Christophe Gagnon-Audet, Touraj Laleh, Mojtaba Faramarzi, Kartik Ahuja, and Irina Rish. "SAND-Mask: An Enhanced Gradient Masking Strategy for the Discovery of Invariances in Domain Generalization." In: *arXiv preprint arXiv:2106.02266* (2021).

[106] Baifeng Shi, Dinghuai Zhang, Qi Dai, Zhanxing Zhu, Yadong Mu, and Jingdong Wang. "Informative Dropout for Robust Representation Learning: A Shape-Bias Perspective." In: *International Conference on Machine Learning*. 2020.

[107] Sahil Singla, Besmira Nushi, Shital Shah, Ece Kamar, and Eric Horvitz. "Understanding Failures of Deep Networks via Robust Feature Extraction." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021.

[108] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting." In: *Journal of Machine Learning Research* (2014).

[109] Tatiana Tommasi, Novi Patricia, Barbara Caputo, and Tinne Tuytelaars. "A Deeper Look at Dataset Bias." In: *German Conference on Pattern Recognition*. 2015.

[110] Shashank Tripathi, Siddhartha Chandra, Amit Agrawal, Ambrish Tyagi, James M Rehg, and Visesh Chari. "Learning to Generate Synthetic Data via Compositing." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019.

[111] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. "Adversarial Discriminative Domain Adaptation." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2017.

[112] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is All You Need." In: *Advances in neural information processing systems* (2017).

[113] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. "Deep Hashing Network for Unsupervised Domain Adaptation." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2017.

[114] Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John C Duchi, Vittorio Murino, and Silvio Savarese. "Generalizing to Unseen Domains via Adversarial Data Augmentation." In: *Advances in Neural Information Processing Systems*. 2018.

[115] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. "Tent: Fully Test-Time Adaptation by Entropy Minimization." In: *International Conference on Learning Representations*. 2021.

[116] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip Yu. "Generalizing to Unseen Domains: A Survey on Domain Generalization." In: *IEEE Transactions on Knowledge and Data Engineering* (2022).

[117] Zijian Wang, Yadan Luo, Ruihong Qiu, Zi Huang, and Mahsa Baktashmotlagh. "Learning to Diversify for Single Domain Generalization." In: *IEEE/CVF International Conference on Computer Vision*. 2021.

[118] Syed Waqas Zamir, Aditya Arora, Akshita Gupta, Salman Khan, Guolei Sun, Fahad Shahbaz Khan, Fan Zhu, Ling Shao, Gui-Song Xia, and Xiang Bai. "Isaid: A Large-Scale Dataset for Instance Segmentation in Aerial Images." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2019.

[119] Gui-Song Xia, Xiang Bai, Jian Ding, Zhen Zhu, Serge Belongie, Jiebo Luo, Mihai Datcu, Marcello Pelillo, and Liangpei Zhang. "DOTA: A Large-Scale Dataset for Object Detection in Aerial Images." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018.

[120] Zhenlin Xu, Deyi Liu, Junlin Yang, Colin Raffel, and Marc Niethammer. "Robust and Generalizable Visual Representation Learning via Random Convolutions." In: *International Conference on Learning Representations*. 2021.

[121] Tao Yang, Shenglong Zhou, Yuwang Wang, Yan Lu, and Nanning Zheng. "Test-time Batch Normalization." In: *arXiv preprint arXiv:2205.10210* (2022).

[122] Haotian Ye, Chuanlong Xie, Tianle Cai, Ruichen Li, Zhenguo Li, and Liwei Wang. "Towards a Theoretical Framework of Out-of-Distribution Generalization." In: *Advances in Neural Information Processing Systems* (2021).

[123] Fuming You, Jingjing Li, and Zhou Zhao. "Test-Time Batch Statistics Calibration for Covariate Shift." In: *arXiv preprint arXiv:2110.04065* (2021).

[124] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. "Cutmix: Regularization Strategy to Train Strong Classifiers with Localizable Features." In: *IEEE/CVF International Conference on Computer Vision*. 2019.

[125] Chongzhi Zhang, Mingyuan Zhang, Shanghang Zhang, Daisheng Jin, Qiang Zhou, Zhongang Cai, Haiyu Zhao, Xianglong Liu, and Ziwei Liu. "Delving Deep into the Generalization of Vision Transformers under Distribution Shifts." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022.

[126] Marvin Mengxin Zhang, Sergey Levine, and Chelsea Finn. "MEMO: Test Time Robustness via Adaptation and Augmentation." In: *NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications*. 2021.

[127] Xingxuan Zhang, Linjun Zhou, Renzhe Xu, Peng Cui, Zheyan Shen, and Haoxin Liu. "Towards Unsupervised Domain Generalization." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022.

[128] Yabin Zhang, Minghan Li, Ruihuang Li, Kui Jia, and Lei Zhang. "Exact Feature Distribution Matching for Arbitrary Style Transfer and Domain Generalization." In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022.

[129] Long Zhao, Ting Liu, Xi Peng, and Dimitris Metaxas. "Maximum-Entropy Adversarial Data Augmentation for Improved Generalization and Robustness." In: *Advances in Neural Information Processing Systems* (2020).

dernière page de la thèse

AUTORISATION DE SOUTENANCE

Vu les dispositions de l'arrêté du 25 mai 2016 modifié par l'arrêté du 26 août 2022,

Vu la demande des directeurs de thèse

Messieurs L. CHEN et E. DELLANDREA

et les rapports de

Mme N. VINCENT
Professeure - Université Paris-Cité - Laboratoire LIPADE
Equipe Systèmes Intelligents de Perception - 45, rue des Saints-Pères - 75270 Paris Cedex 06

et de

M. A. HABRARD
Professeur - Université Jean Monnet - Laboratoire Hubert Curien
18 rue du Professeur Benoît Lauras - 42000 Saint-Etienne

Monsieur DUBOUDIN Thomas

est autorisé à soutenir une thèse pour l'obtention du grade de **DOCTEUR**

Ecole doctorale Informatique et Mathématiques

Fait à Ecully, le 12 décembre 2022

Pour le directeur de l'Ecole centrale de Lyon
Le directeur des Formations

Grégory VIAL