Encouraging Intra-Class Diversity Through a Reverse Contrastive Loss for Single-Source Domain Generalization

Thomas Duboudin[1], Emmanuel Dellandréa[1], Corentin Abgrall[2], Gilles Hénaff[2], Liming Chen[1]
[1] LIRIS, École Centrale de Lyon, France
{thomas.duboudin, emmanuel.dellandrea, liming.chen}@ec-lyon.fr
[2] Thales LAS France
{corentin.abgrall, gilles.henaff}@fr.thalesgroup.com

Abstract

Traditional deep learning algorithms often fail to generalize when they are tested outside of the domain of the training data. The issue can be mitigated by using unlabeled data from the target domain at training time, but because data distributions can change dynamically in real-life applications once a learned model is deployed, it is critical to create networks robust to unknown and unforeseen domain shifts. In this paper we focus on one of the reasons behind the inability of neural networks to be so: deep networks focus only on the most obvious, potentially spurious, clues to make their predictions and are blind to useful but slightly less efficient or more complex patterns. This behaviour has been identified and several methods partially addressed the issue. To investigate their effectiveness and limits, we first design a publicly available MNIST-based benchmark to precisely measure the ability of an algorithm to find the "hidden" patterns. Then, we evaluate state-of-the-art algorithms through our benchmark and show that the issue is largely unsolved. Finally, we propose a partially reversed contrastive loss to encourage intra-class diversity and find less strongly correlated patterns, whose efficiency is demonstrated by our experiments.

1. Introduction

While deep neural networks reach or even surpass human-level performance on an increasing number of computer vision tasks, *e.g.*, classification, object detection or semantic segmentation [18, 17], it is found that their performance could drop sharply [22] when mismatch between training and testing data distributions occurs. This is an issue of critical importance for real-world application, where test-time domain shift is common: the data distribution can change over time because of varying factors as diverse as lighting, view angle, sensor, *etc.*, ending up being significantly different from the one used during training. For embedded networks deployed in the wild, that can't be easily retrained (autonomous cars, for instance), geographical distribution change are an issue too. It is impossible to gather enough data to cover all possible shifts, so another solution has to be found.

Figure 1. Samples of our benchmark. First row is training data, second row is testing data. For the training data, all digits are colorized with their class color, without exception. For the testing data, all digits have the same color, which is the average of the colors used in training.

To generalize without having access to any kind of information regarding the target domain is the goal of the research field known as domain generalization (DG) (or *out-of-distribution* generalization). Most of the works in this field assume to have access to data coming from several identified domains during training and aim to make the network's internal representations invariant on the domains, by finding patterns common to all domains. The idea behind is that patterns shared among several domains are more likely to be found in a new, unseen, domain [31, 32]. This multi-source domain generalization (MDG) setting is not realistic for all tasks and all kind of data: health-related data, for instance, cannot be easily shared and the gathering of data from different sources (such as country) may prove difficult. In this paper, we are concerned with a situation, coined as single-source domain generalization (SDG), where only one source domain is available during training, either that all data come from a same distribution, or that the data come from several distri-

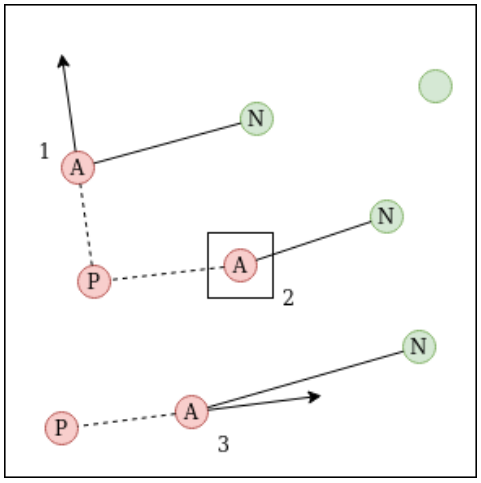

Figure 2. An illustration of our reverse contrastive approach on 3 anchors. Anchor 1 in the latent space is sufficiently far from its closest negative neighbor (full line) for it to be pushed away from its closest positive neighbor (dotted line). So is anchor 3. Anchor 2, however, is too close to a negative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

butions which are not identified differently. Both settings (single and multi-source) are still largely unsolved in the general case: on several complex and realistic datasets, *e.g.*, DomainNet [37], PACS [30], methods designed to improve the out-of-distribution generalization capability of networks were found to perform as badly as the standard training procedure [16].

The reasons behind the failure of deep networks to generalize outside of their training distribution are numerous and complex [33, 39, 19]. Among them is the inability of networks to use several semantically different clues, or patterns, to make their predictions and to instead rely only on the simplest most predictive patterns [40, 20]. If the learned obvious class-specific patterns are spurious and therefore missing (or worse: anti-correlated or uncorrelated with the training class) at test time, the performance of the network will suffer. Furthermore, due to the nature of the data acquisition process, it is fairly common to observe such spurious correlations between patterns and labels in the data [44, 11]. Examples of such biases could be background and context, as in [4], or even inconspicuous hospital specific clues, as in [10]. This behaviour, and its consequences on a network's ability to generalize outside of its training domain, have been previously identified and partially mitigated in [24, 38]. However, using our benchmark specifically crafted to evaluate the ability of a model to find other patterns than the most strongly correlated ones, we show the inability of existing methods, *e.g.*, Jigsaw [5], Spectral Decoupling [38], RSC [24], to correctly handle this issue. None of the existing works reach satisfying performance and some of them can record a performance drop as high as 56 accuracy points when the correlated pattern identified in training data, *i.e.*, color, is missing in the test data. Note that we should not expect the discovery of "hidden" patterns to completely solve the SDG issue on real-life benchmarks, such as on PACS mentionned earlier: a pattern can be missing in the test-time distribution, but also uncorrelated or anti-correlated with its training class. Real-life situations are a complex blend of missing and uncorrelated (or anti-correlated) patterns. In such situations, it is not sufficient to learn the less strongly correlated patterns, some of them have to be ignored for the network to make a proper decision. Similarly, the performance of a DG method on a real-life benchmark is not an assessment of its ability to find the "hidden" patterns. Our benchmark, illustrated in Figure 1 is specifically crafted to avoid this issue.

Given these findings, we posit that, when training for a classification task, naturally learned patterns are learned because they tend to maximize inter-class specificity, while ignoring intra-class variability. These learned patterns are the most effective when it comes to discriminate different classes. As a result, we hypothesize that, to find less efficient 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 don't have access to additional auxiliary labels to discriminate elements within a class, the idea is to expand the class-wise internal representations of a deep network. We do so by using a partially reversed contrastive loss: in a traditional contrastive approach, same class samples are brought together in the latent space while different class samples are pushed away from each other. In our approach, same class samples are pushed away from each others until they reach a different class sample (see Figure 2 for an illustration of our method). Our strategy yields state-of-the-art performance on our MNIST-based dataset.

Our contributions are threefold:

- We design an MNIST-based benchmark, named MNIST-MP (Missing Patterns), to study the ability of an algorithm to find other useful patterns than the naturally learned ones. It is available on github [1].
- We show that several state-of-the-art domain generalization algorithms especially designed to find "hidden" patterns fail to do so on a simple MNIST-based benchmark.

[1] https://github.com/BlindSubmissionAuthor/RCL-MNIST-MP-UP

- We propose a reverse contrastive loss (RCL) to find less strongly correlated patterns by encouraging intra-class diversity.

The remaining of the paper is organized as follows. Sec.2 discusses related works. Sec.3 introduces the proposed MNIST-based benchmark. Sec.4 presents our proposed RCL algorithm. Sec.5 benchmarks our proposed algorithm along with state-of-the-art algorithms for comparison. Sec.6 concludes the paper and draws some future works.

2. Related Works

2.1. Domain Generalization (DG)

2.1.1 Robustness through multi source training

Most settings in DG assume to have data coming from several different source domains, which are identified. By learning domain invariant features, through a great diversity of practical approaches, the model is supposedly able to generalize to a new domain. Most recent works use adversarial strategies: in the work of Tzeng *et al*. [45] the features extracted by a feature 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 were initially developed in the context of domain adaptation (DA) but can be adapted in the multi-source domain generalization setting by employing all the domains at disposal during training, *e.g*., [6] and [31]. The work of Krueger *et al*. [28] is enforcing invariance at loss level by minimizing the variance of the loss over all training domains alongside with the usual average of the loss per batch. Self-supervised strategies have also emerged to solve the problem of multi-source domain generalization. Carlucci *et al*. [5] used a self-supervised strategy based on solving jigsaw puzzles: the images are divided into tiles, which 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 hence generalize better to a new distribution. So far, it is unclear why features learned with self-supervised tasks are more robust to domain shifts.

In 2019, Gulrajani and Lopez-Par [16] analysed a lot of DG algorithms and benchmarks "in search of lost domain generalization". They detailed ways to carefully and realistically evaluate multi-source DG methods, by using larger models, stronger data augmentation, correct model and hyper-parameters selection (without any use of the target domain), and doing all on more datasets. Their conclusion was 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*. [7] who look into the latent spaces from differently trained networks, one with DANN [12] and another with the basic ERM. They have shown that ERM naturally tends to yield domain invariant features.

2.1.2 Robustness through stylization

In situations where only one domain is available during training (SDG), style transfer as a mean of data augmentation has been recently introduced independently by several authors [34, 48, 27, 41] as a way to promote invariance to texture changes. The images are simply stylized differently, with AdaIN based [23] or GAN-based [15, 50] architectures, before being fed to the main task model. It has been hypothesized that the low ability of deep networks to transfer to unseen domains was related to their texture bias: deep networks grant more importance to the texture than the shape in image recognition tasks [13]. As such, correcting this texture bias is expected to improve out-of-distribution generalization, provided that the distribution shift is mainly due to a texture shift. The stylization as data augmentation exists in different flavors: Nam *et al*. [34] use intra-domain extra-class stylization: the styles used to transfer are coming from the images of the training distribution but from different class samples than the image being stylized. Yue *et al*. [48] use several auxiliary style datasets. Jackson *et al*. [25] approximated the style codes distribution of paintings with a multivariate normal, and completely discard the painting dataset once the transfer network is trained. The stylization is a strong baseline in single-source domain generalization (SDG) [48] when the domain shift encountered is a texture shift, such as a synthetic to real transfer, which, unfortunately, does not always hold in the general case.

2.1.3 Robustness by overcoming biases

Deep neural networks learn correlations between patterns and labels no matter how spurious they seem to a human being and therefore sometimes need to be explicitly told not to use certain patterns for the recognition task at hand [35, 26]. In a domain shift situation, the patterns naturally learned by a network are precisely the biases we need to overcome to find other and semantically different patterns. A line of work, *e.g.*, Kim *et al*. [26], is able to create bias invariant features, but requires the bias to be given as an auxiliary label, and are therefore not suitable to the SDG setting where no information on testing data, *e.g.*, bias shift, is assumed available. Another line of work, *e.g.*, [35, 9, 1], focuses on

finding counter-examples *i.e.*, samples that do not share the majority biases of their classes, and increase their importance during training. Assuming the necessary existence of counter-examples is an optimistic hypothesis in certain situations, such as with synthetic data where only a few number of synthetic models of interest may be available. Besides, it could lead to overstate the importance of what could be an aberrant outlier, such as an annotation error. As such, there also exists approaches which do not need counter-examples. Representation Self-Challenging [24] (RSC) introduces a dropout-based strategy, where the feature map coefficients most responsible for the prediction are muted, instead of random ones, therefore leading the network to focus on other less correlated patterns. Spectral Decoupling [38], a method introduced by Pezeshki *et al.*, proposes an L2 regularization on the output logits of the network, with a theoretical motivation, to push the network to learn other patterns than the obvious and spurious ones.

2.2. Supervised Contrastive Learning

Contrastive learning [43, 21] has been introduced in deep learning to enable class-wise manipulation of a network's latent space with a simple idea: elements of the same class should be close to each other in the latent space, elements of different classes should be away from each other in the latent space. Our approach is to loosen the latent space to identify less correlated patterns in a recognition task. While this idea has been previously studied in [49] and [47], it was applied with a different goal. Zhang *et al*. [49] proposes to take all the space available on the unit sphere evenly, but the regularization only impacts negative pairs. Xuan *et al*. [47] aims to create a latent space with better generalization to unseen classes using a modified triplet loss that also pushes together elements of different classes.

3. MNIST "Missing-Patterns"

3.1. Training Set

We introduce a new toy dataset based on the MNIST dataset [29] to benchmark the ability of an algorithm to find less efficient but still useful patterns in the data. For the training data, each digit, coming from the original MNIST training data, is colorized with a color depending on its class. There are no counter-examples (digit colorized differently than the majority of the other digits in its class), as in [1] or label noise (digit colorized with their correct class color, but with an assigned label different from the original one), as in [38]. We do so to study the ability of a network to find more semantically different patterns than what is done naturally, without having any kind of data signals that could help it find those patterns. The validation data is colorized the same way as the training data but come from the original MNIST test split. A sample of the dataset can be found Figure 1. A summary of the synthetic MNIST-based DG datasets published so far is available in Table 1.

Method	SDG/MDG	UP/MP	LN/CE	Classes
IRM [2]	MDG (2)	UP	LN	2
SD [38]	SDG	UP	LN	2
GIP [1]	SDG	UP & MP	CE	10
Ours	SDG	MP	None	10

Table 1. Comparison of MNIST-datasets introduced to study domain generalization strategies. SDG/MDG refers to single or multiple training domains, UP/MP is uncorrelated (or anti-correlated) or missing patterns, LN is label noise, CE is counter-examples, Classes is the number of classes.

3.2. Testing Set

The MNIST-MP dataset aims to benchmark SDG algorithms in presence of missing patterns, *i.e.*, when a class correlated pattern in training, *e.g.*, color, is missing in testing. It is made by coloring each digit with the same color, no matter their class. The images come from the original MNIST test data. The test color and the training colors are not chosen randomly. To precisely confront a deep network with a situation where useful training patterns are missing at test-time, the test 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 and has to make use of other class correlated patterns, *e.g.*, shape. Because we can't directly abide by this constraint we propose to use the average training color as the test color, 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. If the training color of class C is closer to the average color than the other colors, the network will wrongfully believe that test samples are all of class C. If two training colors are close to each other, the network won't be able to easily differentiate samples of the two classes using only the color and will be driven to learn other class related patterns, *e.g.*, shape, at least partially. 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.

4. Reverse Contrastive Approach

4.1. The proposed method

Our method starts from the observation that patterns that are learned with a standard training procedure (*e.g.*, stochas-

Method	Validation Accuracy	Test Accuracy
Standard Training Procedure	99.8 ($\pm$ 0.006)	26.0 ($\pm$ 4.7)
Dropout	99.8 ($\pm$ 0.01)	31.9 ($\pm$ 3.1)
Dropout & Orthogonality [3]	99.8 ($\pm$ 0.008)	42.7 ($\pm$ 2.6)
Dropout & Covariance [8]	99.8 ($\pm$ 0.002)	42.6 ($\pm$ 2.0)
Jigsaw Puzzle [5]	99.8 ($\pm$ 0.01)	43.0 ($\pm$ 3.0)
(with early stopping @95)	98.5 ($\pm$ 0.3)	59.9 ($\pm$ 4.5)
Reconstruction	99.8 ($\pm$ 0.007)	28.5 ($\pm$ 4.8)
Spectral Decoupling [38]	99.8 ($\pm$ 0.004)	47.7 ($\pm$ 2.9)
(with early stopping @95)	99.4 ($\pm$ 0.09)	49.8 ($\pm$ 1.5)
RSC [24]	99.2 ($\pm$ 0.2)	43.5 ($\pm$ 5.0)
RCL-SDG (ours)		
$m = 0.2$	99.8 ($\pm$ 0.007)	12.5 ($\pm$ 2.6)
$m = 0.4$	99.8 ($\pm$ 0.007)	19.9 ($\pm$ 5.5)
$m = 0.6$	99.8 ($\pm$ 0.009)	36.8 ($\pm$ 5.6)
$m = 0.8$	99.7 ($\pm$ 0.03)	55.7 ($\pm$ 4.7)
$m = 0.9$	99.4 ($\pm$ 0.08)	68.2 ($\pm$ 4.0)
(with early stopping @95)	95.84 ($\pm$ 0.6)	74.7 ($\pm$ 8.5)
$\boldsymbol{m = \infty}$	**96.0 ($\pm$ 0.3)**	**89.9 ($\pm$ 0.9)**
Standard Training Procedure on the original MNIST	97.8 ($\pm$ 0.12)	97.8 ($\pm$ 0.12)

Table 2. Results of the methods on the MNIST "Missing-Patterns" dataset. The results are obtained with a best validation selection strategy. The first column is the validation accuracy (colored digits), and the second the test (grey digits) accuracy. Accuracies reported are averages over 10 runs, with the standard deviation between parenthesis. The last line indicates the accuracy reached by our backbone on the original MNIST dataset without domain shift, it thus gives the highest bound of the accuracy in the test domain for the methods.

tic gradient descent with a cross-entropy loss, for a classification task) are learned because they exhibit the maximum inter-class specificity but they ignore 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, a). To find such patterns, we spread the intermediate features (the output of the convolutional features extractor F, before the fully connected classifier layers) of elements of the same class, to a certain extent. The limit in the class-wise repulsion is fixed with 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 constrastive 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 for the features extractor only:

$$\begin{aligned} &min_F\{-d(f_a, f_p)\} \\ &\textit{s.t. } d(f_a, f_p) < m \times d(f_a, f_n) \end{aligned} \tag{1}$$

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. We use a multiplicative margin, instead of an additive margin, contrary to the standard triplet loss [21], to ease the hyper-parameter search. 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):

$$\mathcal{L}_{RC} = \begin{cases} -d(f_a, f_p) \textit{ if } d(f_a, f_p) < m \times d(f_a, f_n) \\ 0 \textit{ otherwise} \end{cases} \tag{2}$$

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 visualisation of the proposed method can be found in Figure 2 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

4.2. State-of-the-art baselines

A first algorithm to find several useful patterns for classification can simply make use of dropout [42]. 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 [3] (more precisely, we apply the double soft orthogonality regularization) and constraint over the covariance matrix of the filters activations, used in [8]. 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 to be close to another. The penalisation 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. This constitutes what we call the naive strategies.

We also implement more elaborated methods inspired by DG works reviewed in sec.2. First, we compare our approach with one inspired from the jigsaw puzzle multi-task strategy used in [5]. Our approach is also compared to a reconstruction multi-task strategy (inspired from [14]), 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 are more than what is extracted with a single classification objective. The two strategies tailored precisely for the problem at hand are Representation Self Challenging [24], and Spectral Decoupling [38]. 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 these using style transfer for instance, because our goal is to measure the ability of an algorithm to find less correlated patterns.

5. Results and Discussion

5.1. Experimental setup

Our experiments are conducted on our benchmark using a small neural network. The architecture we use was introduced in [5] to study MNIST to SVHN [36] 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 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,

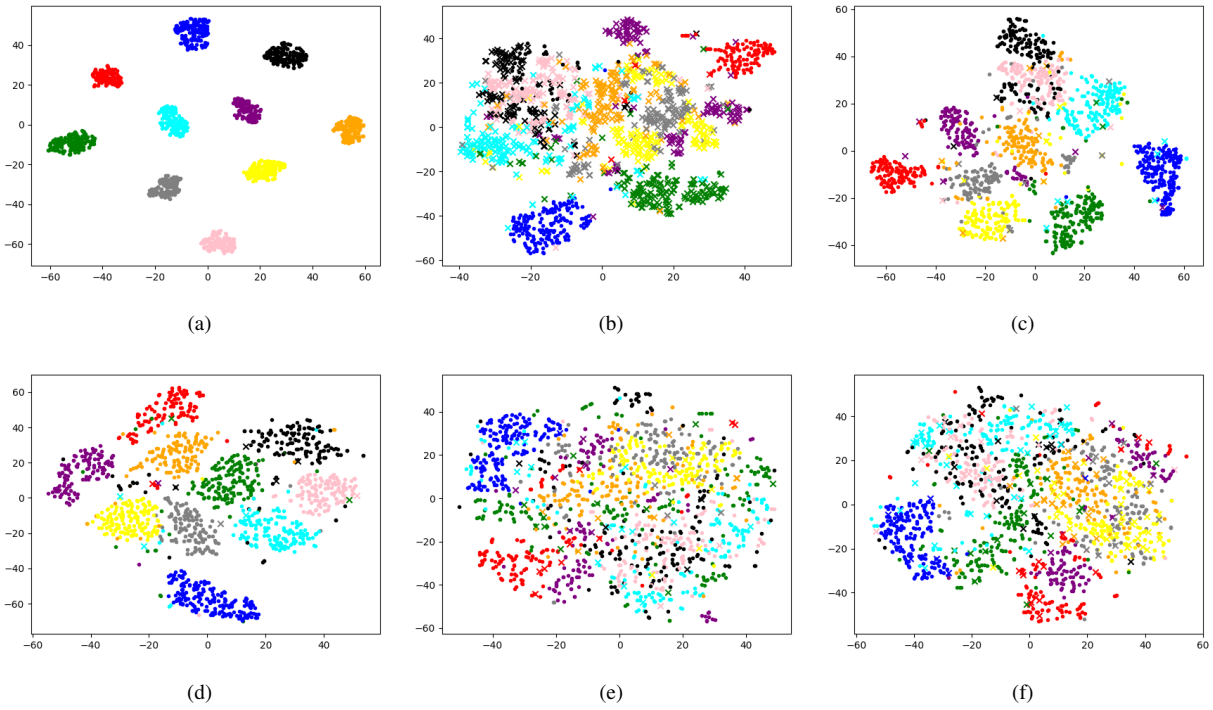

Figure 3. Visualisation of different latent spaces through a 2-dimensional T-SNE [46]. The two axis are the dimensions obtained by the T-SNE. *(a)* shows the validation (colored digits) latent space, for a model trained normally. *(b)* shows the test (grey digits) latent space for the same model. *(c)* 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.

as in [5], 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, permutation with the shuffled batch. The decoder in the reconstruction method is composed of 4 transposed convolution layers, with an 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, reconstruction) 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 behaviour 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 [28]. Therefore, we average the results over 10 runs, and report the standard deviation alongside the average.

5.2. Results and analysis

Table 2 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 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 suffers 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%. 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 to focus only on a subset of useful features are not effective enough to consider the problem solved on our simple dataset.

Beside the accuracy table, we also illustrate the impact of our reverse contrastive loss in Figure 3 by looking at the latent space extracted from models trained differently, through the T-SNE dimensionality reduction method [46] 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 one another only roughly (e). This explains why the performance in the training domain is lower: the classifier can't 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 to find useful patterns. The cluster's sizes needed to obtain an accuracy 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. We hypothesize that on more complex datasets, the RCL without limitation might not yield good results due to the images specificities being more prevalent than in MNIST. To tackle this issue is a future work direction.

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 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).

6. Conclusion

In this paper we carefully created a benchmark to study one of the reasons deep networks fail to generalize outside of their training domain: the reliance of a model on only the most obvious discriminant patterns has dramatic consequences if, in the test domain, such patterns are missing. We showed that existing methods only mitigate the damage. 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 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 benchmark. The goal was not to compete directly against other SDG methods on realistic benchmarks, but to provide a building block for a future general SDG algorithm. The future works will be dedicated to the study of more real-life-like situations, and to the decorrelation (or anti-correlation) issue. Our benchmark MNIST-MP is publicly available on github, together with an MNIST-UP also proposed for the anti-correlation studies.

Acknowledgment

This work was in part supported by the 4D Vision project funded by the Partner University Fund (PUF), a FACE program, as well as Alegoria project and Arès labcom projects, both funded by the French Research Agency, l'Agence Nationale de Recherche (ANR), under grants ANR-17-CE38-0014-03 and ANR-16-LCV2-0012-01, respectively.

References

[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] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2020.

[3] 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.

[4] Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In *IEEE/CVF European conference on computer vision*, 2018.

[5] 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.

[6] 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.

[7] Junbum Cha, Hancheol Cho, Kyungjae Lee, Seunghyun Park, Yunsung Lee, and Sungrae Park. Domain generalization needs stochastic weight averaging for robustness on domain shifts. *arXiv preprint arXiv:2102.08604*, 2021.

[8] Michael Cogswell, Faruk Ahmed, Ross Girshick, Larry Zitnick, and Dhruv Batra. Reducing overfitting in deep networks by decorrelating representations. *Internation Conference on Learning Representations*, 2016.

[9] 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. *arXiv preprint arXiv:2102.06406*, 2021.

[10] Alex J DeGrave, Joseph D Janizek, and Su-In Lee. Ai for radiographic covid-19 detection selects shortcuts over signal. *Nature Machine Intelligence*, 2021.

[11] Simone Fabbrizzi, Symeon Papadopoulos, Eirini Ntoutsi, and Ioannis Kompatsiaris. A survey on bias in visual datasets, 2021.

[12] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, 2015.

[13] 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.

[14] 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.

[15] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2014.

[16] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. In *International Conference on Learning Representations*, 2021.

[17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *IEEE/CVF International Conference on Computer Vision*, 2017.

[18] 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.

[19] Katherine Hermann, Ting Chen, and Simon Kornblith. The origins and prevalence of texture bias in convolutional neural networks. *Advances in Neural Information Processing Systems*, 2020.

[20] Katherine Hermann and Andrew Lampinen. What shapes feature representations? exploring datasets, architectures, and training. *Advances in Neural Information Processing Systems*, 2020.

[21] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International Workshop on Similarity-Based Pattern Recognition*, 2015.

[22] 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.

[23] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *IEEE/CVF International Conference on Computer Vision*, 2017.

[24] 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.

[25] 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.

[26] 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*, June 2019.

[27] Myeongjin Kim and Hyeran Byun. Learning texture invariant representation for domain adaptation of semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[28] 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). *arXiv preprint arXiv:2003.00688*, 2020.

[29] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs. Available: http://yann.lecun.com/exdb/mnist*, 2010.

[30] 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.

[31] 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.

[32] Daniel Moyer, Shuyang Gao, Rob Brekelmans, Aram Galstyan, and Greg Ver Steeg. Invariant representations without adversarial training. In *Advances in Neural Information Processing Systems*, 2018.

[33] Vaishnavh Nagarajan, Anders Andreassen, and Behnam Neyshabur. Understanding the failure modes of out-of-distribution generalization. In *International Conference on Learning Representations*, 2020.

[34] Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. Reducing domain gap via style-agnostic networks. *arXiv preprint arXiv:1910.11645*, 2019.

[35] 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.

[36] 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.

[37] 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.

[38] Mohammad Pezeshki, Sékou-Oumar Kaba, Yoshua Bengio, Aaron Courville, Doina Precup, and Guillaume Lajoie. Gradient starvation: A learning proclivity in neural networks. *arXiv preprint arXiv:2011.09468*, 2020.

[39] 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.

[40] 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.

[41] Nathan Somavarapu, Chih-Yao Ma, and Zsolt Kira. Frustratingly simple domain generalization via image stylization. *arXiv preprint arXiv:2006.11207*, 2020.

[42] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 2014.

[43] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2014.

[44] Tatiana Tommasi, Novi Patricia, Barbara Caputo, and Tinne Tuytelaars. A deeper look at dataset bias. In *German Conference on Pattern Recognition*, 2015.

[45] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017.

[46] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 2008.

[47] Hong Xuan, Abby Stylianou, Xiaotong Liu, and Robert Pless. Hard negative examples are hard, but useful. In *IEEE/CVF European Conference on Computer Vision*, 2020.

[48] Xiangyu Yue, Yang Zhang, Sicheng Zhao, Alberto Sangiovanni-Vincentelli, Kurt Keutzer, and Boqing Gong. Domain randomization and pyramid consistency: Simulation-to-real generalization without accessing target domain data. In *IEEE/CVF International Conference on Computer Vision*, 2019.

[49] Xu Zhang, Felix X. Yu, Sanjiv Kumar, and Shih-Fu Chang. Learning spread-out local feature descriptors. In *IEEE/CVF International Conference on Computer Vision*, 2017.

[50] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE/CVF International Conference on Computer Vision*, 2017.