Episodic Training for Domain Generalization

Citation for published version:

Digital Object Identifier (DOI):
10.1109/ICCV.2019.00153

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
2019 IEEE/CVF International Conference on Computer Vision (ICCV)

General rights
Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact openaccess@ed.ac.uk providing details, and we will remove access to the work immediately and investigate your claim.
Domain generalization (DG) is the challenging and topical problem of learning models that generalize to novel testing domains with different statistics than a set of known training domains. The simple approach of aggregating data from all source domains and training a single deep neural network end-to-end on all the data provides a surprisingly strong baseline that surpasses many prior published methods. In this paper we build on this strong baseline by designing an episodic training procedure that trains a single deep network in a way that exposes it to the domain shift that characterizes a novel domain at runtime. Specifically, we decompose a deep network into feature extractor and classifier components, and then train each component by simulating it interacting with a partner who is badly tuned for the current domain. This makes both components more robust, ultimately leading to our networks producing state-of-the-art performance on three DG benchmarks. Furthermore, we consider the pervasive workflow of using an ImageNet trained CNN as a fixed feature extractor for downstream recognition tasks. Using the Visual Decathlon benchmark, we demonstrate that our episodic-DG training improves the performance of such a general purpose feature extractor by explicitly training a feature for robustness to novel problems. This shows that DG training can benefit standard practice in computer vision.

1. Introduction

Machine learning methods often degrade rapidly in performance if they are applied to domains with very different statistics to the data used to train them. This is the problem of domain shift, which domain adaptation (DA) aims to address in the case where some labelled or unlabelled data from the target domain is available for adaptation [2, 36, 22, 10, 23, 4]; and domain generalisation (DG) aims to address in the case where no adaptation to the target problem is possible [27, 12, 18, 33] due to lack of data or computation. DG is a particularly challenging problem setting, since explicit training on the target is disallowed; yet it is particularly valuable due to its lack of assumptions. For example, it would be valuable to have a domain-general visual feature extractor that performs well ‘out of the box’ as a representation for any novel problem, even without fine-tuning.

The significance of the DG challenge has led to many studies in the literature. These span robust feature space learning [27, 12], model architectures that are purpose designed to enable robustness to domain shift [16, 39, 17] and specially designed learning algorithms for optimising standard architectures [33, 18] that aim to fit them to a more robust minima. Among all these efforts, it turns out that the naive approach [17] of aggregating all the training domains’ data together and training a single deep network end-to-end is very competitive with state-of-the-art, and better than many published methods – while simultaneously being much simpler and faster than more elaborate alternatives. In this paper we aim to build on the strength and simplicity of this simple data aggregation strategy, but improve it by designing an episodic training scheme to improve DG.

The paradigm of episodic training has recently been popularised in the area of few-shot learning [9, 28, 34]. In this problem, the goal is to use a large amount of background source data, to train a model that is capable of few-shot learning when adapting to a novel target problem. However despite the data availability, training on all the source data would not be reflective of the target few-shot learning condition. So in order to train the model in a way that reflects how it will be tested, multiple few-shot learning training episodes are setup among all the source datasets [9, 28, 34].

How can an episodic training approach be designed for domain generalisation? Our insight is that, from the perspective of any layer $l$ in a neural network, being exposed to a novel domain at testing-time is experienced as that layer’s neighbours $l-1$ or $l+1$ being badly tuned for the problem at hand. That is, neighbours provide input to the current layer (or accept output from it) with different statistics to the current layer’s expectation.
Therefore to design episodes for DG, we should expose layers to neighbours that are untrained for the current domain. If a layer can be trained to perform well in this situation of badly tuned neighbours, then its robustness to domain-shift has increased.

To realise our episodic training idea, we break networks up into feature extractor and classifier modules and train them with our episodic framework. This leads to more robust modules that together obtain state-of-the-art results on several DG benchmarks. Our approach benefits from end-to-end learning, while being model agnostic (architecture independent), and simple and fast to train; in contrast to most existing DG techniques that rely on non-standard architectures [17], auxiliary models [33], or non-standard optimizers [18].

Finally, we provide a practical demonstration of the value of explicit DG training, beyond the isolated benchmarks that are common in the literature. Specifically, we consider whether DG can benefit the common practitioner workflow of using an ImageNet [31] pre-trained CNN as a feature extractor for novel tasks and datasets. The standard (homogeneous) DG problem setting assumes shared label-spaces between source and target domain, thus highly restricting its applicability. To benefit the wider computer vision workflow, we go beyond this to heterogeneous DG (Table 5). That is, to train a feature extractor specifically to improve its robustness in representing novel downstream tasks without fine-tuning. Using the Visual Decathlon benchmark [29], we show that Episodic training provides an improved representation for novel downstream tasks compared to the standard ImageNet pre-trained CNN.

2. Related Work

Multi-Domain Learning (MDL) MDL aims to learn several domains simultaneously using a single model [3, 29, 30, 40]. Depending on the problem, how much data is available per domain, and how similar the domains are, multi-domain learning can improve [40] – or sometimes worsen [3, 29, 30] – performance compared to a single model per domain. MDL is related to DG because the typical setting for DG is to assume a similar setup in that multiple source domains are provided. But that now the goal is to learn how to extract a domain-agnostic or domain-robust model from all those source domains. The most rigorous benchmark for MDL is the Visual Decathlon (VD) [29]. We repurpose this benchmark for DG by training a CNN on a subset of the VD domains, and then evaluating its performance as a feature extractor on an unseen disjoint subset of them. We are the first to demonstrate DG at this scale, and in the heterogeneous label setting required for VD.

Domain Generalization Despite different details, previous DG methods can be divided into a few categories by motivating intuition. Domain Invariant Features: These aim to learn a domain-invariant feature representation, typically by minimising the discrepancy between all source domains – and assuming that the resulting source-domain invariant feature will work well for the target as well. To this end [27] employed maximum mean discrepancy (MMD), while [12] proposed a multi-domain reconstruction auto-encoder to learn this domain-invariant feature. More recently, [20] applied MMD constraints within the representation learning of an autoencoder via adversarial training. Hierarchical Models: These learn a hierarchical set of model parameters, so that the model for each domain is parameterised by a combination of a domain-agnostic and a domain-specific parameter [16, 17]. After learning such a hierarchical model structure on the source domains the domain agnostic parameter can be extracted as the model with the least domain-specific bias, that is most likely to work on a target problem. This intuition has been exploited in both shallow [16] and deep [17] settings. Data Augmentation: A few studies proposed data augmentation strategies to synthesise additional training data to improve the robustness of a model to novel domains. These include the Bayesian network [33], which perturbs input data based on the domain classification signal from an auxiliary domain classifier. Meanwhile, [37] proposed an adversarial data augmentation method to synthesize ‘hard’ data for the training model to enhance its generalization. Optimisation Algorithms: A final category of approach is to modify a conventional learning algorithm in an attempt to find a more robust minima during training, for example through meta-learning [18]. Our approach is different to all of these in that it trains a standard deep model, without special data augmentation and with a conventional optimiser. The key idea requires only a simple modification of the training procedure to introduce appropriately constructed episodes. Finally, in contrast to the small datasets considered previously, we demonstrate the impact of DG model training in the large scale VD benchmark.

Neural Network Meta-Learning Learning-to-learn and meta-learning methods have resurfaced recently, in particular in few-shot recognition [9, 34, 25], and learning-to-optimize [28] tasks. Despite significant other differences in motivation and methodological formalisations, a common feature of these methods is an episodic training strategy. In few-shot learning, the intuition is that while lot of source tasks and data may be available, these should be used for training in a way that closely simulates the testing condition. Therefore at each learning iteration, a random subset of source tasks and instances are sampled to generate a training episode defined by a random few-shot learning task of similar data volume and cardinality as the model is expected to be tested on at runtime. Thus the model eventually ‘sees’ all the training data in aggregate, but in any given iteration, it is evaluated in a condition similar to a real ‘testing’ condition. In this paper we aim to develop an episodic training strategy to improve domain-robustness, rather than learning-to-learn. While the high-level idea of an episodic strategy is the same, the DG problem and associated episode construction details are completely different.
3. Methodology

In this section we will first introduce the basic dataset aggregation method (AGG) which provides a strong baseline for DG performance, and then subsequently present three episodic training strategies for training it more robustly.

Problem Setting. In the DG setting, we assume that we are given $n$ source domains $D = \{D_1, \ldots, D_n\}$, where $D_i$ is the $i^{th}$ source domain containing data-label pairs $(x^i_j, y^i_j)$. The goal is to use these to learn a model $f : x \rightarrow y$ that generalises well to a novel testing domain $D_\ast$ with different statistics to the training domains, without assuming any knowledge of the testing domain during model learning.

For homogeneous DG, we assume that all the source domains and the target domain share the same label space $Y_i = Y_\ast = Y_\ast$, $\forall i, j \in [1, n]$. For the more challenging heterogeneous setting, the domains can have different, potentially completely disjoint label spaces $Y_i \neq Y_\ast \neq Y_j$. We will start by introducing the homogeneous case and discuss the heterogeneous case later.

Architecture. We break neural network classifiers $f : x \rightarrow y$ into a sequence modules. In practice, we use two: A feature extractor $\theta(\cdot)$ and a classifier $\psi(\cdot)$, so that $f(x) = \psi(\theta(x))$.

3.1. Overview

Vanilla Aggregation Method. A simple approach to the DG problem is to simply aggregate all the source domains’ data together, and train a single CNN end-to-end ignoring the domain label information entirely [17]. This approach is simple, fast and competitive with more elaborate state-of-the-art alternatives. In terms of neural network modules, it means that both the classifier $\psi$ and the feature extractor $\theta$ are shared across all domains\(^1\), as illustrated in Fig. 1, leading to the objective function:

$$\arg\min_{\theta, \psi} \mathbb{E}_{D_i \sim D} \left[ \mathbb{E}_{(x, y) \sim D_i} \left[ \ell(y, \psi(\theta(x))) \right] \right]$$

(1)

where $\ell(\cdot)$ is the cross-entropy loss here.

Domain Specific Models. Our goal is to improve robustness by exposing individual modules to neighbours that are badly calibrated to a given domain. To obtain these ‘badly calibrated’ components, we also train domain-specific models. As illustrated in Fig. 2, this means that each domain $i$ has its own model composed of feature extractor $\theta_i$ and classifier $\psi_i$. Each domain-specific module is only exposed to data of that corresponding domain. To train domain-specific models, we optimise:

$$\arg\min_{[\theta_1, \ldots, \theta_n], [\psi_1, \ldots, \psi_n]} \mathbb{E}_{D_i \sim D} \left[ \mathbb{E}_{(x, y) \sim D_i} \left[ \ell(y, \psi_i(\theta_i(x))) \right] \right]$$

(2)

Episodic Training. Our goal is to train a domain agnostic model, as per $\psi$ and $\theta$ in the aggregation method in Eq. 1. And we will design an episodic scheme that makes use of the domain-specific modules as per Eq. 2 to help the domain-agnostic model achieve the desired robustness. Specifically, we will generate episodes where each domain agnostic module $\psi$ and $\theta$ is paired with a domain-specific partner that is mismatched with the current data being input. So module and data combinations of the form $(\psi, \theta, x_i')$ and $(\psi_j, \theta, x_i')$ where $i \neq i'$.

3.2. Episodic Training of Feature Extractor

To train a robust feature extractor $\theta$, we ask it to learn features which are robust enough that data from domain $i$ can be processed by a classifier that has never experienced domain $i$ before as shown in Fig. 3. To generate episodes according to this criterion, we optimise

$$\arg\min_{\theta} \mathbb{E}_{i,j \sim [1, n], i \neq j} \left[ \mathbb{E}_{(x, y) \sim D_i} \left[ \ell(y, \psi_j(\theta(x))) \right] \right]$$

(3)

where $i \neq j$ and $\psi_j$ means that $\psi_j$ is considered constant for the generation of this loss, i.e., it does not receive back-propagated gradients. This gradient-blocking is important, because without it the data $x_i$ from domain $i$ would ‘pollute’ the classifier $\psi_j$ which we want to retain as being naive to domains outside of $j$.

Thus in this optimisation, only the feature extractor $\theta$ is penalized whenever the classifier $\psi_j$ makes the wrong prediction. That means that, for this loss to be minimised, the shared feature extractor $\theta$ must map data $x_i$ into a format that a ‘naive’ classifier $\psi_j$ can correctly classify. The feature extractor must learn to help a classifier recognize a data point that is from a domain that is novel to that classifier.

---

\(^1\) $i$ indicates domain index and $j$ indicates instance number within domain.

For simplicity, we will omit $j$ in the following.

\(^2\) At least in the homogeneous case.
3.3. Episodic Training of Classifier

Analogous to the above, we can also interpret DG as the requirement that a classifier should be robust enough to classify data even if it is encoded by a feature extractor which has never seen this type of data in the past, as illustrated in Fig. 3. Thus to train the robust classifier \(\psi\) we ask it to classify domain \(i\) instances \(x_i\), fed through a domain \(j\)-specific feature extractor \(\theta_j\). To generate episodes according to this criterion, we do:

\[
\arg\min_{\psi} \mathbb{E}_{i,j \sim [1,n], i \neq j} [\mathbb{E}_{(x_i,y_i) \sim D_i}[\ell(y_i, \psi(\theta_j(x_i)))]]
\]

where \(i \neq j\) and \(\theta_j\) means \(\theta_j\) is considered constant for generation of the loss here. Similar to the feature extractor module, this operation is important to retain the domain-specificity of feature extractor \(\theta_j\). The result is that only the classifier \(\psi\) is penalised, and in order to minimise this loss \(\psi\) must be robust enough to accept data \(x_i\) that has been encoded by a naive feature extractor \(\theta_j\).

3.4. Episodic Training by Random Classifier

The episodic feature training strategy above is limited to the homogeneous DG setting, since it requires all domains to share label-space in order to create episodes. But in the heterogeneous scenarios, the shared label-space assumption is not met. We next introduce a novel feature training strategy that is suitable for both homogeneous and heterogeneous label-spaces.

In Section 3.2, we introduced the notion of regularising a deep feature extractor by requiring it to support a classifier inexperienced with data from the current domain. Taking this to an extreme, we consider asking the feature extractor to support the predictions of a classifier with random weights, as shown in Fig. 4. To this end, our objective function here is:

\[
\arg\min_{\theta} \mathbb{E}_{D \sim D} \left[\mathbb{E}_{(x,y) \sim D} \left[\ell(y, \psi(\theta(x)))\right]\right]
\]

where, \(\psi\) is a randomly initialised classifier, and \(\overrightarrow{\psi}\) means it is a constant not updated in the optimization. This can be seen as an extreme version of our earlier episodic cross-domain feature extractor training (not only it has not seen any data from domain \(x_i\), but it has not seen any data at all). Moreover, it has the benefit of not requiring a label-space to be shared across all training domains unlike the previous method in Eq. 3.

Specifically, in Eq. 3, the routing \(\psi_i \rightarrow \theta \rightarrow \psi_j\) requires \(\psi_j\) to have a label-space matching \((x_i,y_j)\). But for Eq. 5, each domain can be equipped with its own random classifier \(\psi_r\), with a cardinality matching its normal label-space. This property makes Eq. 5 suitable for heterogeneous domains.

3.5. Algorithm Flow

Our full algorithm brings together the domain agnostic modules that are our goal to train and the supporting domain-specific modules that help train them (Section 3.1). We generate episodes according to the three strategies introduced above. Referring the losses in Eq. 1, 2, 3, 4, 5 as \(L_{agg}, L_{ds}, L_{epic}, L_{epif}, L_{epir}\), then overall we optimise:

\[
L_{full} = L_{agg} + L_{ds} + \lambda_1 L_{epif} + \lambda_2 L_{epic} + \lambda_3 L_{epir}
\]

for parameters \(\theta, \phi, \{\theta_i, \psi_i\}_{i=1}^n\). The full pseudocode for the algorithm is given in Algorithm 1. It is noteworthy that, in practice, when training we first warm up the domain-specific branches for a few iterations before training both the domain-specific and domain-agnostic modules jointly. After training, only the domain agnostic modules (of AGG) will be deployed for testing.
4. Experiments

4.1. Datasets and Settings

Datasets We evaluate our algorithm on three different homogeneous DG benchmarks and introduce a novel and larger scale heterogeneous DG benchmark. The datasets are: IXMAS [38] is cross-view action recognition task. Two object recognition benchmarks include: VLCS [8], which includes images from four famous datasets PASCAL VOC2007 (V) [7], LabelMe (L) [32], Caltech (C) [19] and SUN09 (S) [6] and the more recent PACS which has a larger cross-domain gap than VLCS [17]. It contains four domains covering Photo (P), Art Painting (A), Cartoon (C) and Sketch (S) images. VLCS contains four domains covering Photo (P), Art Painting (A), Car- toon (C) and Sketch (S) images.

Competitors We evaluate the following competitors: AGG the vanilla aggregation method, introduced in Eq. 1, trains a single model for all source domains. DICA [27] a kernel-based method for learning domain invariant feature representations. LRE-SVM [39] a SVM-based method, that trains different SVM model for each source domain. For a test domain, it uses the SVM model from the most similar source domain. D-MTAE [12] a de-noising multi-task auto encoder method, which learns domain invariant features by cross-domain reconstruction. DSN [4] Domain Separation Networks decompose the sources domains into shared and private spaces and learns them with a reconstruction task. TF-CNN [17] learns a domain-agnostic model by factoring out the common component from a set of domain-specific models, as well as tensor factorization to compress the model parameters. CCSSA [26] uses semantic alignment to regularize the learned feature subspace. DANN [11] Domain Adversarial Neural Networks train a feature extractor with a domain-adversarial loss among the source domains. The source-domain invariant feature extractor is assumed to generalise better to novel target domains. MAML [9] The model-agnostic meta-learning method for fast adaptation, repurposed for DG. MLDG [18] A recent meta-learning based optimization method. It mimics the DG setting by splitting source domains into meta-train and meta-test, and modifies the optimisation to improve meta-test performance.

Fusion [24] A method that fuses the predictions from source domain classifiers for the target domain. MMD-AAE [20] A recent method that learns domain invariant feature autoencoding with adversarial training and ensuring that domains are aligned with adversarial training and ensuring that domains are aligned.

4.2. Evaluation on IXMAS dataset

Settings IXMAS contains 11 different human actions. All actions were video recorded by 5 cameras with different views (referred as 0,...,4). The goal is to train an action recognition model on a set of source views (domains), and recognise the action from a novel target view (domain). We follow [20] to keep the first 5 actions and use the same Dense trajectory features as input. For our method, we follow [20] to use a one-hidden layer network with 2000 hidden neurons as our backbone and report the average result of 20 runs. The optimizer is M-SGD with learning rate 1e-4, momentum 0.9, weight decay 5e-5. We use λ₁=2.0, λ₂=2.0, and λ₃=0.5.

Results From the results in Table 1, we can see that: (i) The vanilla aggregation method, AGG is a strong competitor compared to several prior published methods, as is DANN, which is newly identified by us as an effective DG algorithm. (ii) Overall our Epi-FCR performs best, improving 2.4% on AGG, and 1.1% on prior state-of-the-art MMD-AAE. (iii) Particularly in view 1&2 our method achieves new state-of-the-art performance.

4.3. Evaluation on VLCS dataset

Settings VLCS domains share 5 categories: bird, car, chair, dog and person. We use pre-extracted DeCAF6 features and follow [26] to randomly split each domain into train (70%) and test (30%) and do leave-one-out evaluation. We use a 2 fully connected layer architecture with output size of 1024 and 128 with ReLU activation, as per [26] and report the average performance of 20 trials. The optimizer is M-SGD with learning rate 1e-3, momentum 0.9 and weight decay 5e-5. We use λ₁=7.0, λ₂=5.0, and λ₃=0.5.

Results From the results in Table 2, we can see that: (i) The simple AGG baseline is again competitive with many published alternatives, so is DANN. (ii) Our Epi-FCR method achieves the best performance, improving on AGG by 1.7% and performing comparably to prior state-of-the-art MMD-AAE and MLDG with 0.6% improvement over both.

4.4. Evaluation on PACS dataset

Settings PACS is a recent dataset with different object style depictions, and a more challenging domain shift than VLCS, as shown in [17]. This dataset shares 7 object categories across domains, including dog, elephant, giraffe, guitar, horse, horse and person. We follow the protocol in [17] including the recommended train and validation split for fair comparison. We first follow [17] in using the ImageNet pretrained AlexNet (in Table 3) and subsequently also use a modern ImageNet pre-trained ResNet-18 (in Table 4) as a base CNN architecture. We

Ablated variants such as Epi-F denote feature regularisation alone, etc. Episodic is implemented using PyTorch 3.

train our network using the M-SGD optimizer (batch size/per domain=32, lr=1e-3, momentum=0.9, weight decay=5e-5) for 45k iterations when using AlexNet and train our network using the same optimizer (weight decay=1e-4) for ResNet-18. We use $\lambda_1=2.0, \lambda_2=0.05$, and $\lambda_3=0.1$ for both settings. We use the official PACS protocol and split [17] and rerun MetaReg [1] on this split, since MetaReg did not release their protocol.

**Results** From the AlexNet results in Table 3, we can see that: (i) Our episodic method obtained the best performance on held out domains C and S and comparable performance on A, P domains. (ii) It also achieves the best performance overall, with 3.3% improvement on vanilla AGG, and at least 1.7% improvement on prior state-of-the-art methods MLDG [18], Fusion [24] and MetaReg [1].

Meanwhile in Table 4, we see that with a modern ResNet-18 architecture, the basic results are improved across the board as expected. However our full episodic method maintains the best performance overall, with a 2.4% improvement on AGG.

We note here that when using modern architectures like [35, 13] for DG tasks we need to be careful with batch normalization [14]. Batchnorm accumulates statistics of the training data during training, for use at testing. In DG, the source and target domains have domain-shift between them, so different ways of employing batch norm produce different results. We tried two ways of coping with batch norm, one is directly using frozen pre-trained ImageNet statistics. Another is to unfreeze and accumulate statistics from the source domains. We observed that when training ResNet-18 on PACS with accumulating the statistics from source domains it produced a worse accuracy than freezing ImageNet statistics (75.7% vs 79.1%).

### 4.5. Further Analysis and Insights

**Ablation Study** To understand the contribution of each component of our model, we perform an ablation study using PACS-AlexNet shown in Fig. 6a. Episodic training for the feature extractor, gives a 1.6% boost over the vanilla AGG. Including episodic training of the classifier, further improves 0.5%. Finally, combine all the episodic training components, provides

![Figure 5: Cross-domain test accuracy on PACS (AlexNet) with shared feature extractor or classifier. $A\rightarrow C$ means, feed A data through C-specific module. Eg, left: $x_A \mapsto \theta \mapsto \psi_C$, right: $x_A \mapsto \theta_C \mapsto \psi$.](image)

3.3% improvement over vanilla AGG. This confirms that each component of our model contributes to final performance.

**Cross-Domain Testing Analysis** To understand how our Epi-FCR method obtains its improved robustness to domain shift, we study its impact on cross-domain testing. Recall that when we activate the episodic training of the agnostic feature extractor and classifier, we benefit from the domain specific branches by routing domain $i$ data across domain $j$ branches. E.g., we feed: $x_i \mapsto \theta \mapsto \psi_j \mapsto y_j$ to train Eq. 3, and $x_i \mapsto \theta_j \mapsto \psi_i \mapsto y_i$ to train Eq. 4.

Therefore it is natural to evaluate cross-domain testing after training the models. As illustrated in Fig. 5, we can see that the episodic training strategy indeed improves cross-domain testing performance. For example, when we feed domain $A$ data to domain $C$ classifier $x_A \mapsto \theta \mapsto \psi_C \mapsto y_A$, the Episodic-trained agnostic extractor $\theta$ improves the performance of the domain-C classifier who has never experienced domain A data (Fig. 5, left); and similarly for the Episodic-trained agnostic classifier.

**Analysis of Solution Robustness** In the above experiments we confirmed that our episodic model outperforms the strong AGG baseline in a variety of benchmarks, and that each component of our framework contributes. In terms of analysing the mechanism by which episodic training improves robustness to domain shift, one possible route is through leading the model to find a higher quality minima. Several studies recently have analysed learning algorithm variants in terms of the quality of the minima that they lead a model to [15, 5].

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0,1,2,3,4</td>
<td>61.5</td>
<td>75.8</td>
<td>78.0</td>
<td>75.8</td>
<td>79.1</td>
<td>75.0</td>
<td>70.7</td>
<td>71.6</td>
<td>74.2</td>
<td>73.1</td>
<td>76.9</td>
</tr>
<tr>
<td>0,1,2,3,4</td>
<td>72.5</td>
<td>86.9</td>
<td>92.3</td>
<td>92.3</td>
<td>94.5</td>
<td>94.1</td>
<td>93.6</td>
<td>93.8</td>
<td>94.0</td>
<td>94.2</td>
<td>94.8</td>
</tr>
<tr>
<td>0,1,3,4</td>
<td>74.7</td>
<td>84.5</td>
<td>91.2</td>
<td>94.5</td>
<td>95.6</td>
<td>97.3</td>
<td>95.7</td>
<td>95.9</td>
<td>97.5</td>
<td>95.7</td>
<td>99.0</td>
</tr>
<tr>
<td>0,2,3,4</td>
<td>67.0</td>
<td>83.4</td>
<td>90.1</td>
<td>91.2</td>
<td>93.4</td>
<td>95.4</td>
<td>95.4</td>
<td>94.2</td>
<td>97.0</td>
<td>95.7</td>
<td>98.0</td>
</tr>
<tr>
<td>1,2,3,4</td>
<td>71.4</td>
<td>92.3</td>
<td>93.4</td>
<td>93.4</td>
<td>95.7</td>
<td>95.7</td>
<td>93.6</td>
<td>94.0</td>
<td>94.7</td>
<td>94.4</td>
<td>96.3</td>
</tr>
</tbody>
</table>

| Ave. | 69.4 | 84.6 | 87.0 | 90.1 | 91.9 | 91.5 | 90.2 | 89.9 | 91.4 | 90.6 | 93.0 |

Table 1: Cross-view action recognition results (accuracy, %) on IXMAS dataset. Best result in bold.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>L,C,S</td>
<td>63.7</td>
<td>60.6</td>
<td>63.9</td>
<td>67.1</td>
<td>67.7</td>
<td>66.4</td>
<td>67.7</td>
<td>65.5</td>
<td>65.0</td>
<td>65.4</td>
<td>67.1</td>
</tr>
<tr>
<td>V,C,S</td>
<td>58.2</td>
<td>59.7</td>
<td>60.1</td>
<td>62.1</td>
<td>62.6</td>
<td>64.0</td>
<td>61.3</td>
<td>60.0</td>
<td>60.2</td>
<td>60.6</td>
<td>64.3</td>
</tr>
<tr>
<td>V,L,S</td>
<td>79.7</td>
<td>88.1</td>
<td>89.1</td>
<td>92.3</td>
<td>94.4</td>
<td>92.6</td>
<td>94.4</td>
<td>92.0</td>
<td>92.3</td>
<td>92.3</td>
<td>94.1</td>
</tr>
<tr>
<td>V,L,C</td>
<td>61.0</td>
<td>54.9</td>
<td>61.3</td>
<td>59.1</td>
<td>64.4</td>
<td>63.6</td>
<td>65.9</td>
<td>64.7</td>
<td>64.2</td>
<td>65.8</td>
<td>65.9</td>
</tr>
</tbody>
</table>

| Ave. | 65.7 | 65.8 | 68.6 | 70.2 | 72.3 | 71.7 | 72.1 | 70.5 | 70.4 | 71.2 | 72.9 |

Table 2: Cross-domain object recognition results (accuracy, %) on VLCS benchmark. Best in bold.
Table 3: Cross-domain object recognition results (accuracy, %) of different methods on PACS using pretrained AlexNet. Best in bold.

<table>
<thead>
<tr>
<th>Source Target</th>
<th>AGG</th>
<th>MAML</th>
<th>DANN</th>
<th>MetaReg</th>
<th>Epi-C</th>
<th>Epi-F</th>
<th>Epi-R</th>
<th>Epi-FCR</th>
</tr>
</thead>
<tbody>
<tr>
<td>C.PS A</td>
<td>73.9</td>
<td>73.5</td>
<td>79.4</td>
<td>79.1</td>
<td>82.3</td>
<td>82.0</td>
<td>82.1</td>
<td>82.1</td>
</tr>
<tr>
<td>A.PS C</td>
<td>74.9</td>
<td>73.8</td>
<td>77.5</td>
<td>77.8</td>
<td>77.8</td>
<td>77.0</td>
<td>77.5</td>
<td>77.5</td>
</tr>
<tr>
<td>A.C.P S</td>
<td>74.4</td>
<td>74.0</td>
<td>75.1</td>
<td>74.3</td>
<td>94.5</td>
<td>94.3</td>
<td>95.9</td>
<td>95.9</td>
</tr>
<tr>
<td>Ave.</td>
<td>74.1</td>
<td>73.8</td>
<td>76.6</td>
<td>77.1</td>
<td>77.1</td>
<td>76.0</td>
<td>77.0</td>
<td>77.0</td>
</tr>
</tbody>
</table>

Table 4: Cross-domain object recognition results (accuracy, %) of different methods on PACS using ResNet-18. Best in bold.

<table>
<thead>
<tr>
<th>Source</th>
<th>Target</th>
<th>AGG</th>
<th>DANN</th>
<th>MetaReg</th>
<th>Epi-C</th>
<th>Epi-F</th>
<th>Epi-R</th>
<th>Epi-FCR</th>
</tr>
</thead>
<tbody>
<tr>
<td>C.PS</td>
<td>A</td>
<td>64.6</td>
<td>60.3</td>
<td>61.1</td>
<td>62.9</td>
<td>64.1</td>
<td>63.2</td>
<td>66.2</td>
</tr>
<tr>
<td>A.PS</td>
<td>C</td>
<td>64.5</td>
<td>58.7</td>
<td>66.5</td>
<td>67.0</td>
<td>66.8</td>
<td>67.5</td>
<td>66.9</td>
</tr>
<tr>
<td>A.C.P</td>
<td>S</td>
<td>51.1</td>
<td>47.9</td>
<td>58.6</td>
<td>57.5</td>
<td>60.1</td>
<td>57.0</td>
<td>59.0</td>
</tr>
<tr>
<td>Ave.</td>
<td></td>
<td>60.8</td>
<td>64.5</td>
<td>67.4</td>
<td>69.2</td>
<td>70.3</td>
<td>69.0</td>
<td>70.0</td>
</tr>
</tbody>
</table>

Table 5: Cross-domain object recognition results (accuracy, %) of different methods on PACS using pretrained VGG-16. Best in bold.

<table>
<thead>
<tr>
<th>Source Target</th>
<th>AGG</th>
<th>MAML</th>
<th>DANN</th>
<th>MetaReg</th>
<th>Epi-C</th>
<th>Epi-F</th>
<th>Epi-R</th>
<th>Epi-FCR</th>
</tr>
</thead>
<tbody>
<tr>
<td>C.PS A</td>
<td>64.6</td>
<td>60.3</td>
<td>61.1</td>
<td>62.9</td>
<td>64.1</td>
<td>63.2</td>
<td>66.2</td>
<td>61.0</td>
</tr>
<tr>
<td>A.PS C</td>
<td>64.5</td>
<td>58.7</td>
<td>66.5</td>
<td>67.0</td>
<td>66.8</td>
<td>67.5</td>
<td>66.9</td>
<td>67.2</td>
</tr>
<tr>
<td>A.C.P S</td>
<td>51.1</td>
<td>47.9</td>
<td>58.6</td>
<td>57.5</td>
<td>60.1</td>
<td>57.0</td>
<td>59.0</td>
<td>59.0</td>
</tr>
<tr>
<td>Ave.</td>
<td>60.8</td>
<td>64.5</td>
<td>67.4</td>
<td>69.2</td>
<td>70.3</td>
<td>69.0</td>
<td>70.0</td>
<td>67.9</td>
</tr>
</tbody>
</table>

4.6. Evaluation on VD-DG dataset

Heterogeneous Problem Setting Visual Decathlon contains ten datasets and was initially proposed as a multi-domain learning benchmark [29]. We re-purpose Decathlon for a more ambitious challenge of domain generalisation. As explained earlier, our motivation is find out if DG learning can improve the defacto standard ‘ImageNet trained CNN feature extractor’ for use as a fixed off-the-shelf representation for new target problems. In this case the feature extractor is trained on the source domain, and used to extract features of the target domain data. Then a target domain-specific classifier (we use SVM) is trained to classify in the target domain. As explained in Table 5 (left), this is quite different from the standard DG setting in that target domain labels are used (for shallow classifier training), but the focus here is on the robustness of the learned feature when generalising to represent new domains and tasks without further fine-tuning. If DG training can improve feature generalisation compared to a vanilla ImageNet CNN, this could be of major practical value given the widespread usage of this workflow by vision practitioners.

Besides evaluating a potentially more generally useful problem setting compared to standard homogeneous DG, our VD experiment is also a larger scale evaluation compared to existing DG studies. As shown in Table 5 (right), VD-DG has twice the domains of VLCS and PACS and is an order of magnitude larger in terms of data and category numbers.

Settings We consider five larger datasets in VD (CIFAR-100, Daimler Ped, GTSRB, Omniglot and SVHN) as our source domains, and the four smallest datasets (Aircraft, D. Textures, VGG-Flowers and UCF101) as our target domains. The goal is to use DG training among the source datasets to learn a feature which outperforms the off-the-shelf ImageNet-trained CNN that we use as an initial condition. We use ResNet-18 [13] as the backbone model, and resize all images to 64×64 for computational efficiency. To support the VD heterogeneous label space, we assume a shared feature extractor, and a source
Table 5: Left: Difference between conventional homogeneous DG setting and new heterogeneous DG setting. Right: Comparing the larger scale of our VD-DG (excluding ImageNet) vs previous DG benchmarks.

Table 6: Results of top-1 accuracy (%) and visual decathlon overall scores of different methods on VD-DG. Train on CIFAR-100, Daimler Ped, GTSRB, Omniglot, SVHN, and optionally ImageNet (Combine). Test on Aircraft, D. Textures, VGG-Flowers, UCF101.

5. Conclusion

We addressed the domain generalisation problem by proposing a simple episodic training strategy that mimics train-test domain-shift during training, thus improving the trained model’s robustness to novel domains. We showed that our method achieves state-of-the-art performance on all the main existing DG benchmarks. We also performed the largest DG evaluation to date, using the Visual Decathlon benchmark. Importantly, we provided the first demonstration of DG’s potential value ‘in the wild’ — by demonstrating our model’s potential to improve the performance of the defacto standard ImageNet pre-trained CNN as a fixed feature extractor for novel downstream problems.

---

4 We note one concurrent study of the heterogeneous DG setting [21] considered the VD-DG benchmark that we propose here. Their results are slightly higher due do use of a larger image size and cross-validation of SVM parameters (we use sklearn defaults).
References

[16] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Deeper, broader and artier domain generalization. In IJCV, 2017. 1, 2, 3, 5, 6, 7
[17] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Learning to generalize: Meta-learning for domain generalization. In AAAI, 2018. 1, 2, 5, 6, 7, 8
[26] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In ICMRL, 2013. 1, 2, 5, 6, 7
[34] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for deep learning. In CVPR, 2015. 2

