Episodic Training for Domain Generalization

Da Li[1,2], Jianshu Zhang[3], Yongxin Yang[2]
Cong Liu[4], Yi-Zhe Song[2] and Timothy M. Hospedales[1,2,5]

[1]Samsung AI Center, Cambridge [2]SketchX, CVSSP, University of Surrey
[3]University of Science and Technology of China [4]iFlytek Research [5]University of Edinburgh

{d.li, yongxin.yang, y.song}@surrey.ac.uk, xysszjs@mail.ustc.edu.cn
congliu2@iflytek.com, t.hospedales@ed.ac.uk

Abstract

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 characterises 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 resurged recently, in particular in few-shot recognition [9, 34, 25], and learning-to-optimize [28] tasks. Despite signifiant 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.

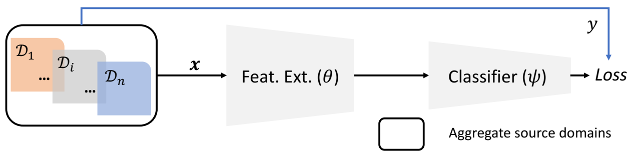

Figure 1: Illustration of vanilla domain-aggregation for multi-domain learning. A single model $\psi(\theta(\cdot))$ classifies data from all domains.

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 $\mathcal{D}=[\mathcal{D}_1,...,\mathcal{D}_n]$, where $\mathcal{D}_i$ is the i^{th} source domain containing data-label pairs $(\mathbf{x}_i^j,y_i^j)$[1]. The goal is to use these to learn a model $f:\mathbf{x}\rightarrow y$ that generalises well to a novel testing domain $\mathcal{D}_*$ 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 $\mathcal{Y}_i=\mathcal{Y}_j=\mathcal{Y}_*$, $\forall i,j\in[1,n]$. For the more challenging *heterogeneous* setting, the domains can have different, potentially completely disjoint label spaces $\mathcal{Y}_i\neq\mathcal{Y}_j\neq\mathcal{Y}_*$. We will start by introducing the homogeneous case and discuss the heterogeneous case later.

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

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 ψ and the feature extractor θ are shared across all domains[2], as illustrated in Fig. 1, leading to the objective function:

$$\underset{\theta,\psi}{\operatorname{argmin}}\ \mathbb{E}_{\mathcal{D}_i\sim\mathcal{D}}\left[\mathbb{E}_{(\mathbf{x}_i,y_i)\sim\mathcal{D}_i}\left[\ell(y_i,\psi(\theta(\mathbf{x}_i))\right]\right] \tag{1}$$

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

[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

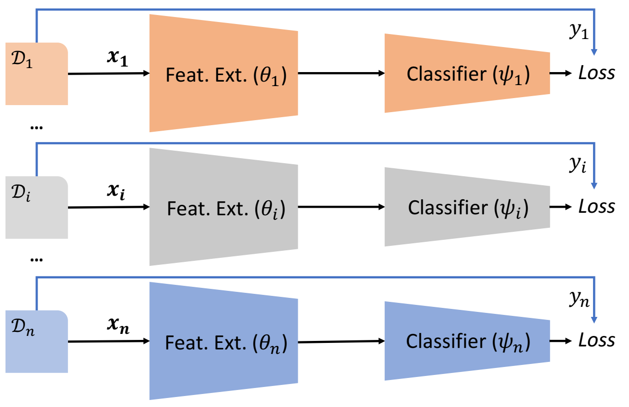

Figure 2: Illustration of domain-specific branches. One classifier and feature extractor are trained per-domain.

composed of feature extractor θ_i and classifier ψ_i. Each domain-specific module is only exposed to data of that corresponding domain. To train domain-specific models, we optimise:

$$\underset{[\theta_1,...,\theta_n],[\psi_1,...,\psi_n]}{\operatorname{argmin}}\ \mathbb{E}_{\mathcal{D}_i\sim\mathcal{D}}\left[\mathbb{E}_{(\mathbf{x}_i,y_i)\sim\mathcal{D}_i}\left[\ell(y_i,\psi_i(\theta_i(\mathbf{x}_i))\right]\right] \tag{2}$$

Episodic Training Our goal is to train a domain agnostic model, as per ψ and θ 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 ψ and θ is paired with a domain-specific partner that is *mismatched* with the current data being input. So module and data combinations of the form $(\psi,\theta_i,x_{i'})$ and $(\psi_i,\theta,x_{i'})$ where $i\neq i'$.

3.2. Episodic Training of Feature Extractor

To train a robust feature extractor θ, 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

$$\underset{\theta}{\operatorname{argmin}}\ \mathbb{E}_{i,j\sim[1,n],i\neq j}\left[\mathbb{E}_{(\mathbf{x}_i,y_i)\sim\mathcal{D}_i}\left[\ell(y_i,\overline{\psi}_j(\theta(\mathbf{x}_i))\right]\right] \tag{3}$$

where $i\neq j$ and $\overline{\psi}_j$ means that ψ_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 $\mathbf{x}_i$ from domain i would 'pollute' the classifier ψ_j which we want to retain as being naive to domains outside of j.

Thus in this optimisation, only the feature extractor θ is penalized whenever the classifier ψ_j makes the wrong prediction. That means that, for this loss to be minimised, the shared feature extractor θ must map data $\mathbf{x}_i$ into a format that a 'naive' classifier ψ_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.

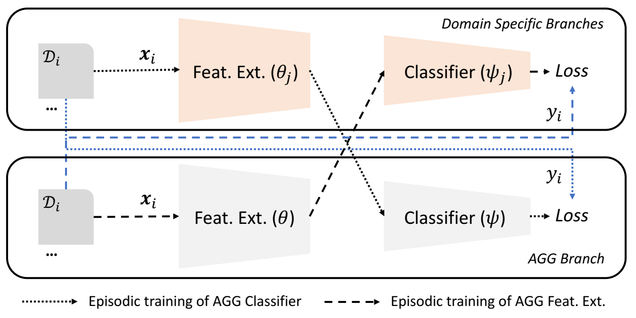

Figure 3: Episodic training for feature and classifier regularisation. The shared feature extractor feeds domain specific classifiers. The shared classifier reads domain-specific feature extractors.

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 ψ we ask it to classify domain i instances $\mathbf{x}_i$ fed through a domain j-specific feature extractor θ_j. To generate episodes according to this criterion, we do:

$$\underset{\psi}{\operatorname{argmin}}\ \mathbb{E}_{i,j\sim[1,n],i\neq j}\left[\mathbb{E}_{(\mathbf{x}_i,y_i)\sim\mathcal{D}_i}\left[\ell(y_i,\psi(\bar{\theta}_j(\mathbf{x}_i))\right]\right] \quad (4)$$

where $i \neq j$ and $\bar{\theta}_j$ means θ_j is considered constant for generation of the loss here. Similar to the training of the feature extractor module, this operation is important to retain the domain-specificity of feature extractor θ_j. The result is that only the classifier ψ is penalised, and in order to minimise this loss ψ must be robust enough to accept data $\mathbf{x}_i$ that has been encoded by a naive feature extractor θ_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:

$$\underset{\theta}{\operatorname{argmin}}\ \mathbb{E}_{\mathcal{D}_i\sim\mathcal{D}}\left[\mathbb{E}_{(\mathbf{x}_i,y_i)\sim\mathcal{D}_i}\left[\ell(y_i,\bar{\psi}_r(\theta(\mathbf{x}_i))\right]\right] \quad (5)$$

where, ψ_r is a randomly initialised classifier, and $\bar{\psi}_r$ 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 $\mathbf{x}_i$, 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 $\mathbf{x}_i \mapsto \theta \mapsto \psi_j$ requires ψ_j to have a label-space matching $(\mathbf{x}_i, y_j)$. But for Eq. 5, each domain can be equipped with its own random classifier ψ_r with a cardinality matching its normal label-space. This property makes Eq. 5 suitable for heterogeneous domains.

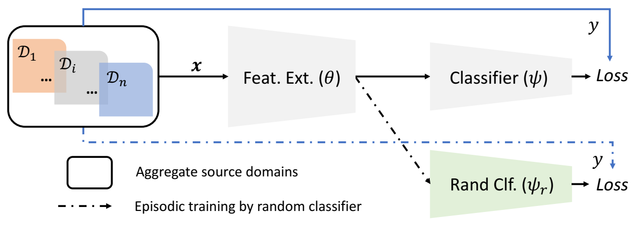

Figure 4: The architecture of random classifier regularization.

Algorithm 1 Episodic Training for Domain Generalization

```
1:  Input: $\mathcal{D}=[\mathcal{D}_1,\mathcal{D}_2,...,\mathcal{D}_n]$
2:  Initialise hyper parameters: $\lambda_1,\lambda_2,\lambda_3,\alpha$
3:  Initialise model parameters: domain specific modules
    $\theta_1, ..., \theta_n$ and $\psi_1, ..., \psi_n$; AGG modules $\theta, \psi$; random
    classifier $\psi_r$
4:  while not done training do
5:      for $(\theta_i,\psi_i)\in[(\theta_1,\psi_1),...,(\theta_n,\psi_n)]$ do
6:          Update $\theta_i := \theta_i - \alpha\nabla_{\theta_i}(L_{ds})$
7:          Update $\psi_i := \psi_i - \alpha\nabla_{\psi_i}(L_{ds})$
8:      end for
9:      Update $\theta := \theta - \alpha\nabla_{\theta}(L_{agg}+\lambda_1 L_{epif}+\lambda_3 L_{epir})$
10:     Update $\psi := \psi - \alpha\nabla_{\psi}(L_{agg}+\lambda_2 L_{epic})$
11: end while
12: Output: $\theta,\psi$
```

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_{epif}, L_{epic}, L_{epir}, then overall we optimise:

$$L_{full} = L_{agg} + L_{ds} + \lambda_1 L_{epif} + \lambda_2 L_{epic} + \lambda_3 L_{epir} \quad (6)$$

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. **VD**: For the final benchmark we repurpose the Visual Decathlon [29] benchmark to evaluate DG.

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 signal. **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. **CCSA** [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 by the MMD constraint. **CrossGrad** [33] A recent method that uses Bayesian networks to perturb the input manifold for DG. **MetaReg** [1] A recent DG method that meta-learns the classifier regularizer. We note that DANN (domain adaptation) is not designed for DG. However, DANN learns domain invariant features, which is natural for DG. And we found it effective for this problem. Therefore we repurpose it as a baseline.

We call our method as **Episodic**. We use **Epi-FCR** to denote our full method with (f)eature regularisation, (c)lassifier regularisation and (r)andom classifier regularisation respectively. Ablated variants such as **Epi-F** denote feature regularisation alone, etc. Episodic is implemented using PyTorch [3].

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 λ_1=2.0, λ_2=2.0, and λ_3=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 λ_1=7.0, λ_2=5.0, and λ_3=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, house, 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 pretrained ResNet-18 (in Table 4) as a base CNN architecture. We

[3]https://github.com/HAHA-DL/Episodic-DG

Source	Target	DICA [27]	LRE-SVM [39]	D-MTAE [12]	CCSA [26]	MMD-AAE [20]	DANN [11]	MLDG [18]	CrossGrad [33]	MetaReg [1]	AGG	Epi-FCR
0,1,2,3	4	61.5	75.8	78.0	75.8	**79.1**	75.0	70.7	71.6	74.2	73.1	76.9
0,1,2,4	3	72.5	86.9	92.3	92.3	94.5	94.1	93.6	93.8	94.0	94.2	**94.8**
0,1,3,4	2	74.7	84.5	91.2	94.5	95.6	97.3	97.5	95.7	96.9	95.7	**99.0**
0,2,3,4	1	67.0	83.4	90.1	91.2	93.4	95.4	95.4	94.2	97.0	95.7	**98.0**
1,2,3,4	0	71.4	92.3	93.4	**96.7**	**96.7**	95.7	93.6	94.0	94.7	94.4	96.3
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.

Source	Target	DICA [27]	LRE-SVM [39]	D-MTAE [12]	CCSA [26]	MMD-AAE [20]	DANN [11]	MLDG [18]	CrossGrad [33]	MetaReg [1]	AGG	Epi-FCR
L,C,S	V	63.7	60.6	63.9	67.1	**67.7**	66.4	67.7	65.5	65.0	65.4	67.1
V,C,S	L	58.2	59.7	60.1	62.1	62.6	64.0	61.3	60.0	60.2	60.6	**64.3**
V,L,S	C	79.7	88.1	89.1	92.3	**94.4**	92.6	**94.4**	92.0	92.3	93.1	94.1
V,L,C	S	61.0	54.9	61.3	59.1	64.4	63.6	**65.9**	64.7	64.2	65.8	**65.9**
Ave.		65.7	65.8	68.6	70.2	72.3	71.7	72.3	70.5	70.4	71.2	**72.9**

Table 2: Cross-dataset object recognition results (accuracy. %) on VLCS benchmark. Best in bold.

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 λ_1=2.0, λ_2=0.05, and λ_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

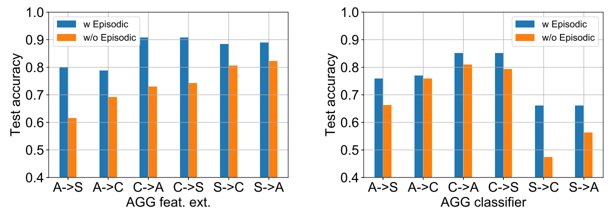

Figure 5: Cross-domain test accuracy on PACS (AlexNet) with shared feature extractor or classifier. A$\mapsto$C means, feed A data through C-specific module. Eg, left: $\mathbf{x}_A \mapsto \theta \mapsto \psi_C$, right: $\mathbf{x}_A \mapsto \theta_C \mapsto \psi$.

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: $\mathbf{x}_i \mapsto \theta \mapsto \psi_j \mapsto y_i$ to train Eq. 3, and $\mathbf{x}_i \mapsto \theta_j \mapsto \psi \mapsto y_i$ 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 $\mathbf{x}_A \mapsto \theta \mapsto \psi_C \mapsto y_A$, the Episodic-trained agnostic extractor θ 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 leads a model to [15, 5].

Source	Target	DICA [27]	D-MTAE [12]	DSN [4]	TF-CNN [17]	Fusion [24]	DANN [11]	MLDG [18]	CrossGrad [33]	MetaReg [1]	AGG	Epi-FCR
C,P,S	A	64.6	60.3	61.1	62.9	64.1	63.2	**66.2**	61.0	63.5	63.4	64.7
A,P,S	C	64.5	58.7	66.5	67.0	66.8	67.5	66.9	67.2	69.5	66.1	**72.3**
A,C,S	P	**91.8**	91.1	83.3	89.5	90.2	88.1	88.0	87.6	87.4	88.5	86.1
A,C,P	S	51.1	47.9	58.6	57.5	60.1	57.0	59.0	55.9	59.1	56.6	**65.0**
Ave.		68.0	64.5	67.4	69.2	70.3	69.0	70.0	67.9	69.9	68.7	**72.0**

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

Source	Target	AGG	DANN [11]	MAML [9]	MLDG [18]	CrossGrad [33]	MetaReg [1]	Epi-FCR
C,P,S	A	77.6	81.3	78.3	79.5	78.7	79.5	**82.1**
A,P,S	C	73.9	73.8	76.5	**77.3**	73.3	75.4	77.0
A,C,S	P	94.4	94.0	**95.1**	94.3	94.0	94.3	93.9
A,C,P	S	70.3	**74.3**	72.6	71.5	65.1	72.2	73.0
Ave.		79.1	80.8	80.6	80.7	77.8	80.4	**81.5**

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

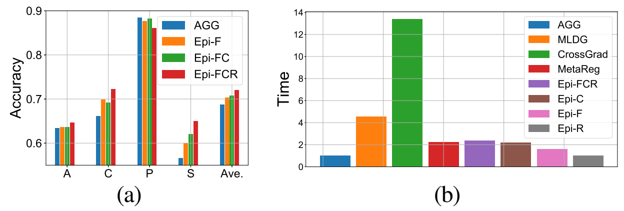

Figure 6: (a) Ablation study on PACS (↑). (b) Computational cost comparison on PACS (↓).

One intuition is that converging to a 'wide' rather than 'sharp' minima provides a more robust solution, because perturbations (such as domain shift, in our case) are less likely to cause a big hit to accuracy if the model's performance is not dependent on a very precisely calibrated solution. Following [15, 41], we therefore compare the solutions found by AGG and our Epi-FCR by adding noise to the weights of the converged model, and observing how quickly the testing accuracy decreases with the magnitude of the noise. From Fig. 7 we can see that both models' performance drops as weights are perturbed, but our Epi-FCR model is more robust to weight perturbations. This suggests that the minima found by Epi-FCR is a more robust one than that found by AGG, which may explain the improved cross domain robustness of Epi-FCR compared to AGG.

Computational Cost Our Episodic model is comparable in cost overall to many contemporaries. Our Epi-C variant does require training multiple feature extractors for the source domains (as do [16, 39, 17, 24]). However, users are more practically interested in testing performance, where our model is as small, fast and simple as AGG (unlike, e.g., [39, 24]). In terms of training requirements, we note that only the Epi-C variant requires multiple feature extractor training, so Epi-FR can still safely be used if this is an issue. Furthermore if a large number of source domains are present, we can sample a subset of them at each batch.

Concretely, we compare the training time of different methods in Fig. 6b. All the methods were run on PACS (ResNet-18) for $3k$ iterations with CPU: Intel i7-7820 (@3.60GHz x 16) and GPU: 1080Ti. As expected vanilla AGG is the fastest to train (9.8 mins), so we regard it as the the base unit. The second tier are our Epi-F and Epi-R. As expected without Epi-C, our Epi-F and Epi-R variants run fast. The next tier are MetaReg, Epi-FCR and MLDG. And the most expensive one is CrossGrad. Although the use of 'Epi-C' here requires domain-specific feature extractors, our Epi-FCR is still comparably efficient. This is because our episodic training does not generate multi-step graph unrolling or meta-optimization in gradient updates. As a result, our time cost is on par with MetaReg [1] and faster than MLDG [18] and CrossGrad [33].

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 evaluation 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

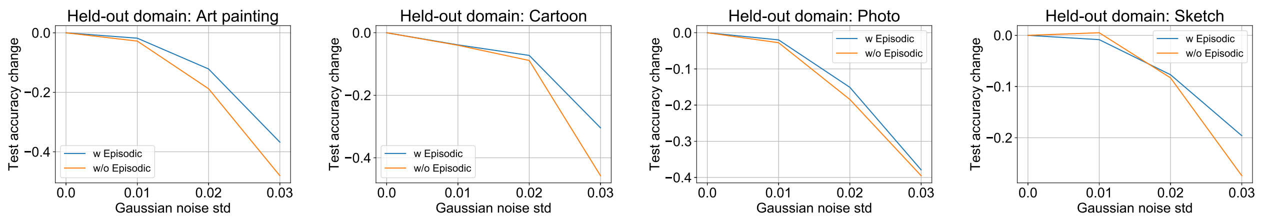

Figure 7: Minima quality analysis: Episodic training (Epi-FCR) vs baseline (AGG).

Setting	Updated in Target Domain? Feature Extractor	Updated in Target Domain? Classifier	Novel Target Labels?
Homogeneous DG	N	N	N
Heterogeneous DG	N	Y	Y

Benchmark	# of data	# of Domains	# of tasks	task space
VLCS	10,729	4	5	Homo.
PACS	9,991	4	7	Homo.
VD-DG	238,215	9	2128	Hetero.

Table 5: Left: Difference between conventional homogeneous DG setting and new heterogeneous DG setting. Right: Contrasting the larger scale of our VD-DG (excluding ImageNet) vs previous DG benchmarks.

Target	ImageNet PT	MLDG [18]			CrossGrad [33]			AGG			DANN [11]			Epi-R		
		Concat	Mean	Combine	Concat	Mean	Combine	Concat	Mean	Combine	Concat	Mean	Combine	Concat	Mean	Combine
Aircraft	12.7	17.4	14.2	15.7	17.2	13.7	15.9	17.4	14.6	15.7	17.4	**15.0**	**16.0**	**17.7**	13.9	15.5
D. Textures	35.2	38.3	34.6	32.5	34.6	31.4	32.2	37.7	35.1	31.5	37.9	36.6	33.0	**40.2**	**37.8**	**33.9**
VGG-Flowers	48.1	54.0	53.2	54.4	49.2	49.3	54.9	**56.3**	52.0	**57.0**	55.5	52.2	53.7	55.4	**53.0**	55.9
UCF101	35.0	44.4	36.7	34.9	42.7	35.7	35.2	43.3	35.0	36.1	44.5	36.1	33.9	**45.7**	**37.1**	**37.3**
Ave.	32.8	38.5	34.7	34.4	35.9	32.5	34.6	38.7	34.2	35.1	38.8	35.0	34.1	**39.7**	**35.5**	**35.7**
VD-Score	185	279	194	169	241	169	169	265	185	172	277	202	165	**304**	**217**	**194**

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.

domain-specific classifier. We perform episodic DG training among the source domains, using our (R)andom classifier model variant, which supports heterogeneous label-spaces. After DG training, the model will then be used as a fixed feature extractor for the held out target domains. With regards to use of ImageNet during training, we consider two settings: (i) Use ImageNet CNN as initial condition, but exclude ImageNet data from DG training, (ii) Include ImageNet as a sixth source domain for DG training. The former helps to constrain training cost, but loses some performance due to the forgetting effect. Therefore we combine (concatenation and mean-pooling) the original ImageNet pre-trained features with the VD-DG trained features. In each case the final feature is used to train a linear SVM for the corresponding task, as per common practice. We train the network using the M-SGD optimizer (batch size/per domain=32, lr=1e-3, momentum=0.9, weight decay=1e-4) for 100k iterations where the lr is decayed in 40k, 80k iterations by a factor 10. We set $\lambda_3 = \frac{2.5}{t+50}$, t is the iteration num.

Results From the results in Table 6, we observed that: (i) All methods use the extra data in VD to improve on the initial features ('ImageNet PT'). (ii) In terms of other DG competitors: Only MLDG, CrossGrad, and DANN were feasible to run on the scale of VD; with others either not supporting heterogeneous label-spaces or scaling to this many domains/examples. (iii) Our Epi-R improves on the strong AGG baseline and DG competitors in both average accuracy, and also the VD score recommended in preference to accuracy in [29]. This demonstrates the value of our Episodic training in learning a feature that is robust to novel domains. (iv) Our concatenation strategy provided the best overall performance compared to directly including ImageNet as a source domain ('Combine'). This partly due to using a fixed 100k iterations to constrain training time. With enough training, the latter option is likely to be best. Overall this is the first demonstration that any DG method can improve robustness to domain shift in a larger-scale setting, across heterogeneous domains, and make a practical impact in surpassing ImageNet feature performance[4].

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

[1] Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. Metareg: Towards domain generalization using meta-regularization. In *NeurIPS*, 2018. 5, 6, 7

[2] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. In *NIPS*, 2006. 1

[3] Hakan Bilen and Andrea Vedaldi. Universal representations: The missing link between faces, text, planktons, and cat breeds. Technical report, 2017. 2

[4] Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. Domain separation networks. In *NIPS*, 2016. 1, 5, 7

[5] Pratik Chaudhar, Anna Choromansk, Stefano Soatt, Yann LeCun, Carlo Baldass, Christian Borg, Jennifer Chays, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. In *ICLR*, 2017. 6

[6] Myung Jin Choi, Joseph Lim, and Antonio Torralba. Exploiting hierarchical context on a large database of object categories. In *CVPR*, 2010. 5

[7] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010. 5

[8] Chen Fang, Ye Xu, and Daniel N. Rockmore. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *ICCV*, 2013. 5

[9] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017. 1, 2, 5, 7

[10] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, 2015. 1

[11] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 2016. 5, 6, 7, 8

[12] Muhammad Ghifary, W. Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. Domain generalization for object recognition with multi-task autoencoders. In *ICCV*, 2015. 1, 2, 5, 6, 7

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6, 7

[14] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 6

[15] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *ICLR*, 2017. 6, 7

[16] Aditya Khosla, Tinghui Zhou, Tomasz Malisiewicz, Alexei Efros, and Antonio Torralba. Undoing the damage of dataset bias. In *ECCV*, 2012. 1, 2, 7

[17] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Deeper, broader and artier domain generalization. In *ICCV*, 2017. 1, 2, 3, 5, 6, 7

[18] 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

[19] Fei-Fei Li, Fergus Rob, and Perona Pietro. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. In *CVPR Workshop on Generative-Model Based Vision*, 2004. 5

[20] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C. Kot. Domain generalization with adversarial feature learning. In *CVPR*, 2018. 2, 5, 6

[21] Yiying Li, Yongxin Yang, Wei Zhou, and Timothy M. Hospedales. Feature-critic networks for heterogeneous domain generalization. In *ICML*, 2019. 8

[22] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *ICML*, 2015. 1

[23] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. Unsupervised domain adaptation with residual transfer networks. In *NIPS*, 2016. 1

[24] Massimiliano Mancini, Samuel Rota Bulo‘, Barbara Caputo, and Elisa Ricci. Best sources forward: Domain generalization through source-specific nets. In *ICIP*, 2018. 5, 6, 7

[25] Nikhil Mishra, Mostafa Rohaninejad, Xi Chen, and Pieter Abbeel. Meta-learning with temporal convolutions. *arXiv*, 2017. 2

[26] Saeid Motiian, Marco Piccirilli, Donald A. Adjeroh, and Gianfranco Doretto. Unified deep supervised domain adaptation and generalization. In *ICCV*, 2017. 5, 6

[27] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *ICML*, 2013. 1, 2, 5, 6, 7

[28] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2017. 1, 2

[29] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Learning multiple visual domains with residual adapters. In *NIPS*, 2017. 2, 5, 7, 8

[30] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Efficient parametrization of multi-domain deep neural networks. In *CVPR*, 2018. 2

[31] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 2

[32] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: A database and web-based tool for image annotation. *IJCV*, 2008. 5

[33] Shiv Shankar, Vihari Piratla, Soumen Chakrabarti, Siddhartha Chaudhuri, Preethi Jyothi, and Sunita Sarawagi. Generalizing across domains via cross-gradient training. In *ICLR*, 2018. 1, 2, 5, 6, 7, 8

[34] Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few shot learning. In *NIPS*, 2017. 1, 2

[35] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016. 6

[36] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv*, 2014. 1

[37] Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John Duchi, Vittorio Murino, and Silvio Savarese. Generalizing to unseen domains via adversarial data augmentation. In *NeurIPS*, 2018. 2

[38] Daniel Weinland, Remi Ronfard, and Edmond Boyer. Free viewpoint action recognition using motion history volumes. *CVIU*, 2006. 5

[39] Zheng Xu, Wen Li, Li Niu, and Dong Xu. Exploiting low-rank structure from latent domains for domain generalization. In *ECCV*, 2014. 1, 5, 6, 7

[40] Yongxin Yang and Timothy M. Hospedales. A unified perspective on multi-domain and multi-task learning. In *ICLR*, 2015. 2

[41] Ying Zhang, Tao Xiang, Timothy M. Hospedales, and Huchuan Lu. Deep mutual learning. In *CVPR*, 2018. 7