BATCH NORMALIZATION EMBEDDINGS FOR DEEP DOMAIN GENERALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

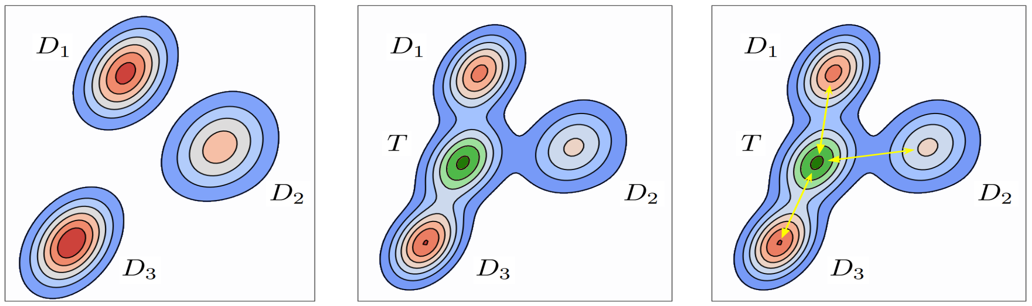
Domain generalization aims at training machine learning models to perform robustly across different and unseen domains. Several recent methods use multiple datasets to train models to extract domain-invariant features, hoping to generalize to unseen domains. Instead, we explicitly train domain-dependant representations by using ad-hoc batch normalization layers to collect independent domain's statistics. We propose to use these statistics to map domains in a shared latent space, where membership to a domain can be measured by means of a distance function. At test time, we project samples from an unknown domain into the same space and express their domain as a linear combination of the known ones. We apply the same mapping strategy at training and test time, learning both a latent representation and a powerful but lightweight ensemble model. We show a significant increase in classification accuracy over current state-of-the-art techniques on popular domain generalization benchmarks: PACS, Office-31 and Office-Caltech.

1 INTRODUCTION

Machine learning models trained on a certain data distribution often fail to generalize to samples from different distributions. This phenomenon is commonly referred to in literature as *domain shift between training and testing data* (Sugiyama & Storkey (2007); Luo et al. (2019)), and is one of the biggest limitations of data driven algorithms. Assuming the availability of few annotated samples from the test domain, the problem can be mitigated by fine-tuning the model with explicit supervision (Yosinski et al. (2014)) or with domain adaptation techniques (Wang & Deng (2018)). Unfortunately, this assumption does not always hold in practice as it is often unfeasible in real scenarios to collect samples for any possible environment.

Domain generalization refers to algorithms to solve the *domain shift* problem by training models robust to unseen domains. Several works leverage different domains at training time to learn a domain-invariant feature extractor (Muandet et al. (2013); Ghifary et al. (2015); Koch et al. (2015); Motiian et al. (2017); Li et al. (2018b)). Other works focus on optimizing the model parameters to obtain consistent performance across domains via ad-hoc training policies (Tobin et al. (2017); Shankar et al. (2018); Volpi et al. (2018); Li et al. (2019)), while a different line of work requires modifications to the model architecture to achieve domain invariance (Khosla et al. (2012); Li et al. (2017); Ding & Fu (2017); Mancini et al. (2018a)).

While these methods try to extract domain-invariant features, we go in the opposite direction and explicitly leverage domain-specific representations by collecting domain-dependent batch normalization (BN) statistics for each of the domains available at training time. By doing so we train a lightweight ensemble of domain-specific models sharing all parameters except for BN statistics. Upon convergence, we propose to use the accumulated statistics to map each domain as a point in a *latent space of domains*, we will refer to this mapping as the Batch Normalization Embedding (BNE) of a domain. Fig. 1 (a) sketches a visualization of such a space for a simplified case of three domains available at training time and a single batch normalization layer operating on the output of a convolutional layer with two output channels (*i.e.,* two means and variances are accumulated and therefore each domain can be represented as a 2D gaussian). Fig. 1 (b, c) illustrate how in this space the membership of a sample to a domain can be measured by looking at the distance (in yellow) between the instance normalization statistics of the sample (in green) and the accumulated population statistics of each domain (in red). The reciprocal of the distances distances are used at test time to



(a) Creation of domain representations.
(b) Projection of a sample from an unknown domain.
(c) Localization with respect to known domains.

Figure 1: We propose to: (a) learn domain specific models and map training domains (in red) in a latent space via population statistics of the BN layers, (b) project a sample from an unknown domain to the same space via its instance statistics (in green), and (c) measure its distance from the known domains to approximate the unknown test distribution by a mixture of the domain models.

classify a test sample from an unknown domain as a weighted combination of the domain-specific predictions of our lightweight ensemble. The same combination of domain-specific models can be used at training time on samples from the known domains to force the ensemble to learn a meaningful latent space and logits that can be linearly combined according to the proposed weighting strategy.

To sum up the contributions of our work: (i) We propose to use batch normalization statistics accumulated on convolutional layers to map image samples to a latent space where membership to a domain can be measured according to a distance from domain BNEs; (ii) We propose an effective way to use this concept to learn a lightweight ensemble model that shares all parameters excepts the normalization statistics and can generalize better to unseen domains; (iii) Compared to previous work, we do not discard domain specific attributes but use them to learn a domain latent space and map unknown domains with respect to known ones. (iv) Our method can be applied to any modern Convolutional Neural Network (CNN) that relies on batch normalization layers and scales gracefully to the number of domains available at training time.

2 Related Work

Domain Generalization. Most domain generalization works attempt to expose the model to domain shift at training time to generalize to target domains. Invariance can be encouraged at multiple levels:

Feature-level, denotes methods deriving domain invariant features by minimizing a discrepancy between multiple training domains. Ghifary et al. (2015) brought domain generalization to the attention of the deep learning community by training multi-task auto-encoders to transform images from one source domain into different ones, thereby learning invariant features. Analogously, Li et al. (2018b) extended adversarial autoencoders by minimizing the Maximum Mean Discrepancy measure to align the distributions of the source domains to an arbitrary prior distribution via adversarial feature learning. Conditional Invariant Adversarial Networks (Li et al. (2018c)) have been proposed to learn domain-invariant representations, whereas Deep Separation Networks (Bousmalis et al. (2016)) extract image representations partitioned into two sub-spaces: one private to each domain and one shared. Differently, Motiian et al. (2017) propose to learn a discriminative embedding subspace via a Siamese architecture (Koch et al. (2015)). Episodic training (Li et al. (2019)) was proposed to train a generic model while exposing it to domain shift. In each episode, a feature extractor is trained with a badly tuned classifier (or vice-versa) to obtain robust features. For all these methods, the limited variety of domains to which the model can be exposed at training time can limit the magnitude of the shift to which the model learns invariance.

Data-level, denotes methods attempting to reduce the training set domain bias by augmenting the cardinality and variety of the samples. Data augmentation methods based on domain-guided perturbations of input samples (Shankar et al. (2018)) or on adversarial examples (Volpi et al. (2018)) have been proposed with the purpose of training a model to be robust to distribution shift. Domain randomization was adopted (Tobin et al. (2017); Loquercio et al. (2019)) to solve the analogous problem of transferring a model from synthetic to real data by extending synthetic data with random renderings. By performing data augmentation those methods force the feature extractor to learn domain-invariant features, while we argue that discarding domain specific information might be detrimental for performance.

Model-based, denotes methods relying on ad-hoc architectures to tackle the domain generalization problem. Li et al. (2017) introduced a low-rank parameterized CNN model, a dynamically parameterized neural network that generalizes the shallow binary undo bias method (Khosla et al. (2012)). Similarly, a structured low-rank constraint is exploited to align multiple domain-specific networks and a domain-invariant one in (Ding & Fu (2017)). Mancini et al. (2018a) train multiple domain-specific classifiers and estimate the probabilities that a target sample belongs to each source domain to fuse the classifiers' predictions. A recent work (Carlucci et al. (2019)) proposes an alternative approach to tackle domain generalization by teaching a model to simultaneously solve jigsaw puzzles and perform well on a task of interest. Most of these methods require changes to state-of-the-art architectures, resulting in an increased number of parameters or complexity of the network.

Meta-learning, denotes methods relying on special training policies to train models robust to domain shift. Li et al. (2018a) extend to domain generalization the widely used model agnostic meta learning framework (Finn et al. (2017)). Balaji et al. (2018) propose a novel regularization function in a meta-learning framework to make the model trained on one domain perform well on another domain. Huang et al. (2020) propose a training heuristic that iteratively discards the dominant features activated on the training data, challenging the model to learn more robust representations. A gradient-based meta-train procedure was introduced by Dou et al. (2019) to expose the optimization to domain shift while regularizing the semantic structure of the feature space. These methods simulate unseen domains by splitting the training data in a meta-training set and meta-test set, therefore are inherently bounded by the variety of the samples available at training time.

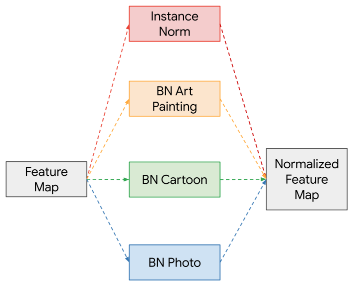
Batch Normalization for distribution alignment. The use of separate batch normalization statistics to align a training distribution to a test one has been firstly introduced for domain adaptation by Carlucci et al. (2017a) and Carlucci et al. (2017b). The same domain-dependent batchnorm layer has been adapted to the multi-domain scenario (Mancini et al. (2018c;b)) and exploited in a graph-based method (Mancini et al. (2019)) that leverages domain meta-data to better align unknown domains to the known ones. In all these works, however, some representation of the target domain is required to perform the alignment during training, using either samples or metadata describing the target domain. Our approach instead does not rely on any external source of information regarding the target domain.

3 METHOD

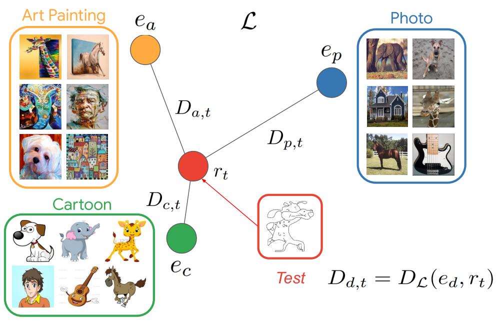
The core idea of our method is to use batch normalization statistics to map known and unknown domains in a shared latent space where domain membership of samples can be measured according to distance between gaussian distributions.

3.1 PROBLEM FORMULATION

Let $\mathcal{X}$ and $\mathcal{Y}$ denote the input (*e.g.,* images) and the output (*e.g.,* object categories) spaces of a model. Let $\mathcal{D} = \{d_i\}_{i=1}^{K}$ denote the set of the K source domains available at training time. Each domain d_i can be described by an unknown conditional probability distribution $p_{x,d_i}^{y} = p(y|x,i)$ over the space $\mathcal{X} \times \mathcal{Y}$. The aim of a machine learning model is to learn the probability distribution $p_x^y = p(y|x)$ of the training set (Bridle (1990)) by training models to learn a mapping $\mathcal{X} \rightarrow \mathcal{Y}$. We propose to use a lightweight ensemble of models to learn a mapping $(\mathcal{X}, \mathcal{D}) \rightarrow \mathcal{Y}$ that leverages the domain label to model a set of conditional distributions $\{p_{x,d_i}^{y}\}_{i=1}^{K}$ conditioned on the domain membership. Let t be a generic target domain available only at testing time and following the unknown probability distribution $p_{x,t}^{y}$ over the same space. Since it is not possible to learn the target distribution $p_{x,t}^{y}$ during training, the goal of our method is to approximate it as a mixture (*i.e.,* linear combination) of the learned source distributions p_{x,d_i}^{y}.



(a) Multi-Source Domain Alignment Layer.



(b) Source domains and target sample projected to $\mathcal{L}$.

Figure 2: Our method on PACS (Li et al. (2017)) with *Sketch* as unknown domain. Our Multi-Source Domain Alignment Layer (a) collects domain-specific population statistics and compute instance statistics for test samples. (b) The population and instance statistics map respectively the source domain and the test sample into the latent space $\mathcal{L}$. Domain similarity is estimated as the reciprocal of the distance $D_{d,t}$ between the target embedding r_t and each domain centroids e_d.

For each source domain $d \in \mathcal{D}$ a training set $\mathcal{S}_d = \{(x_{1_d}, y_{1_d}), ..., (x_{n_d}, y_{n_d})\}$ containing n_d labelled samples is provided. The test set $\mathcal{T} = \{x_{1_t}, ..., x_{n_t}\}$ is composed of m_t unlabelled samples collected from the unknown marginal distribution p_t^x of the target domain t. As opposed to the domain adaptation setting, we assume that samples from the target domain(s) are not available at training time and that each of them might belong to a different unseen domain.

3.2 Multi-Source Domain Alignment Layer

Our work is motivated by the observation that neural networks are particularly prone to capture dataset bias in their internal representations (Li et al. (2016)) making internal features distributions highly domain-dependent. To capture and alleviate the distribution shift that is inherent in the multi-source setting, we draw inspiration from (Carlucci et al. (2017b); Mancini et al. (2018c;b)) and adapt batch normalization layers (Ioffe & Szegedy (2015)) to normalize the domain-dependent activations to the same reference distribution via *domain-specific normalization statistics*. Nevertheless, we differ from previous works since we do not need samples from the target domain to update statistics of our layer and we do not require an extra network which increase the number of parameters (*e.g.,* domain discovery network in (Mancini et al. (2018b))) to compute the domain membership of a sample.

At inference time, the activations of a certain domain d are normalized by matching their first and second order moments, nominally (μ_d, σ_d^2), to those of a reference Gaussian with zero mean and unitary variance:

$$BN(z; d) = \frac{z - \mu_d}{\sqrt{\sigma_d^2 + \epsilon}}, \tag{1}$$

where z is an input activation extracted from the marginal distribution q_d^z of the activations from the domain d; $\mu_d = \mathbb{E}_{z \sim q_d^z}[z]$ and $\sigma_d^2 = Var_{z \sim q_d^z}[z]$ are the population statistics for the domain d, and $\epsilon > 0$ is a small constant to avoid numerical instability.

At training time, the layer collects and applies domain-specific batch statistics $(\tilde{\mu}_d, \tilde{\sigma}_d^2)$, while updating the corresponding moving averages to approximate the domain population statistics. At inference time, if the domain label d of a test sample is unknown or it does not belong to $\mathcal{D}$, we can still rely on normalization by *instance statistics*, *i.e.,* the degenerate case of batch statistics with batch size equal to 1. Fig. 2 (a) depicts the functioning of our multi-source domain generalization layer.

Indeed, for convolutional layers, instance statistics and batch statistics are approximations of the same underlying distribution with different degrees of noise. Since the population statistics are a temporal integration of the batch statistics, the validity of this statement extends to the comparison with them. For example, the computation of the statistics for a single channel in the case of a batch normalization layer applied on a 2D feature map of size $H \times W$ with a generic batch size B (*batch statistics*) and with $B = 1$ (*instance statistics*) is equals to:

$$\tilde{\mu} = \frac{1}{B \cdot H \cdot W} \sum_{b,h,w} z_{b,h,w} \overset{(B=1)}{=} \frac{1}{H \cdot W} \sum_{h,w} z_{h,w} \tag{2}$$

$$\tilde{\sigma}^2 = \frac{1}{B \cdot H \cdot W} \sum_{b,h,w} (z_{b,h,w} - \tilde{\mu})^2 \overset{(B=1)}{=} \frac{1}{H \cdot W} \sum_{h,w} (z_{h,w} - \tilde{\mu})^2, \tag{3}$$

where $\tilde{\mu}$ and $\tilde{\sigma}^2$ are respectively the batch mean and variance and $z_{b,h,w}$ is the value of a single element of the feature map. If we consider $z_{b,h,w}$ to be described by a normally distributed random variable $Z \sim \mathcal{N}(\mu, \sigma^2)$, then the instance and batch statistics are an estimate of the parameters of the same gaussian computed over a different number of samples, $H \cdot W$ and $B \cdot H \cdot W$ respectively. In the next section, we will explain how this property is exploited to map source domains and samples from unknown domain in the same latent space.

3.3 Domain Localization in the Batchnorm Latent Space

The domain alignement layer described in Sec. 3.2 allows to learn the multiple source distributions $\{p^y_{x,d}\}_{d \in \mathcal{D}}$ distinctly. By leveraging them we can learn a lightweight ensemble of domain specific models, where every network shares all the weights except for the normalization statistics. Since such lightweight ensemble embodies the multiple source distributions, we propose to reduce the domain shift on the unknown target domain by interpolating across these distributions to approximate the unknown distribution $p^y_{x,t}$. The choice of the weights depends on the *similarity* of a test sample to each source domain.

We denote with a $l \in \mathcal{B} = \{1, 2, ..., L\}$ in superscript notation the different batch normalization layers in the model. For each of them we can define a latent space $\mathcal{L}^l$ spanned by the activation statistics at the $l - th$ layer of the model. In this space, single samples x are mapped via their instance statics $(\tilde{\mu}, \tilde{\sigma}^2)$, whereas the population statistics accumulated for each domain (μ_d, σ_d^2) are used to represent domain centroids. Considering all latent spaces at different layers we define a batch normalization embedding (BNE) for a certain domain d as the stacking of the population statistics computed at every layer:

$$e_d = [e^1_d, e^2_d, ..., e^L_d] = [(\mu^1_d, {\sigma^1_d}^2), (\mu^2_d, {\sigma^2_d}^2), ..., (\mu^L_d, {\sigma^L_d}^2)]. \tag{4}$$

For a target sample x_t from an unknown domain t, we can derive its projection by forward propagating it through the network and computing its instance statistics. The latent embedding r_t of x_t is defined as the stacked vector of its instance statistics at different batch normalization layers in the network:

$$r_t = [r^1_t, r^2_t, ..., r^L_t] = [(\mu^1_t, {\sigma^1_t}^2), (\mu^2_t, {\sigma^2_t}^2), ..., (\mu^L_t, {\sigma^L_t}^2)]. \tag{5}$$

Each r^l_t represents the instance statistics collected at a certain layer l during forward propagation and can be used to map the sample x_t in the latent space $\mathcal{L}^l$ of layer l. Once the BNE for the test sample is available, it is possible to measure the *similarity* of a target sample x_t to one of the known domains d as the inverse of the distance between r_t and e_d. By extension, this allows a soft 1-Nearest Neighbour domain classification of any test sample.

To compute a distance between two points in $\mathcal{L}^l$, we consider the means and variances of the corresponding batch normalization layer as the parameters of a multivariate Gaussian distribution. We can hence adopt a distance on the space of probability measures, *i.e.,* a symmetric and positive definite function that satisfies the triangle inequality. We select the *Wasserstein distance* for the special case of two multivariate gaussian distributions, but we report a comparison to alternative distances in the supplementary material. Let $p \sim \mathcal{N}(\mu_p, C_p)$ and $q \sim \mathcal{N}(\mu_q, C_q)$ be two normal distributions on R^n, with expected value μ_p and $\mu_q \in R^n$ respectively and C_p, $C_q \in R^{n \times n}$ covariance matrices. Denoting with $|| \cdot ||_2$ the Euclidean norm on R^n, the *2-Wasserstein distance* is:

$$\mathcal{W}(p, q) = \phi((\mu_p, C_p), (\mu_q, C_q)) = ||\mu_p - \mu_q||^2_2 + trace(C_p + C_q - 2(C_q^{1/2} C_p C_q^{1/2})^{1/2}) \tag{6}$$

We rely on Eq. 6 to measure the distance between a test sample x_t and the domain d by summing over the batch normalization layers $l \in \mathcal{B}$ the distance between the activation embeddings r_t^l and e_d^l:

$$D_{\mathcal{L}}(e_d, r_t) = \sum_{l \in \mathcal{B}} \mathcal{W}(e_d^l, r_t^l) = \sum_{l \in \mathcal{B}} \phi((\mu_d^l, Diag({\sigma_d^l}^2)), (\mu_{x_t}^l, Diag({\sigma_{x_t}^l}^2))) \tag{7}$$

Eq. 2 shows that instance and batch statistics differ only for the number of samples over which they are estimated, making the comparison meaningful. The *similarity* of a test sample x_t to the domain d is defined as the reciprocal of the distance from that domain and denoted as w_d^t.

Once the similarity to each source domains is computed, we can use them to recover the unknown target distribution $p_{x,t}^y$ as a mixture (*i.e.,* a linear combination) of the learned source distributions $p_{x,d}^y$ weighted by the corresponding domain similarity:

$$p_{x,t}^y = \frac{\sum_{d \in \mathcal{D}} w_d^t p_{x,d}^y}{\sum_{d \in \mathcal{D}} w_d^t}. \tag{8}$$

We denote with $f(\cdot)$ the result of a forward pass in a neural network. We get the final prediction of our lightweight ensemble model $f(x_t)$ as a linear combination of the domain dependant models $f(x_t|d)$:

$$f(x_t) = \frac{\sum_{d \in \mathcal{D}} w_d^t f(x_t|d)}{\sum_{d \in \mathcal{D}} w_d^t}, \tag{9}$$

Fig. 2 shows an application of our method to the PACS dataset (Li et al. (2017)) with 3 domain available at training time and one unknown.

Our formulation allows to navigate in the latent space of the batchnorm statistics. Specifically, if a test sample belongs to one of the source domains, our method assigns a high weight to the prediction of the corresponding domain specific model. On the other hand, if the test sample does not belong to any of the source domains, the final prediction will be expressed as a linear combination of the models of the domains embodied in our lightweight ensemble.

3.4 Training Policy

To help shape the latent space of every batch normalization layer, we replicate at training time the distance weighting procedure described in Eq. 9 to compute predictions on samples from known domains. Each training batch is composed of K domain batches with an equal number of samples. During every training step, (i) the domain batches are first propagated to update the corresponding domain population statistics $(\mu_d, \sigma_d^2,)$. Then, (ii) all individual samples are propagated assuming an unknown domain to collect their instance statistics and compute the domain similarities w_d^t, as in Sec. 3.3. Finally, (iii) each sample is propagated under K different domain assumptions (*i.e.,* through the corresponding domain-specific branches) and the resulting domain-specific predictions are weighted according to Eq. 9. Applying this procedure during training encourages the creation of a well-defined batch normalization latent space.

Since we initialize our model with weights pre-trained on ImageNet, each domain-specific batch normalization branch needs to be specialized before starting the distance training (DT) procedure described above, otherwise convergence problems might occur. We thus *warm-up* domain-specific batch normalization statistics by pre-training the model on the whole dataset following the standard procedure, except for the accumulation and application of domain-specific batch normalization statistics.

4 Experiments

4.1 Experimental Settings

By means of a synecdoche, we name our method after BNE, its main component.

Datasets. In the main paper we evaluate on two domain generalization benchmarks: (a.) *PACS* (Li et al. (2017)) features 4 domains (*Art Painting*, *Cartoon*, *Photo*, *Sketch*) with a significant domain

Table 1: State-of-the-art comparison on PACS with ResNet-18.

Method	Art	Cartoon	Photo	Sketch	Avg. DA	Avg.	Δ%
CrossGrad - Shankar et al. (2018)	78.7	73.3	94.0	65.1	79.1	77.8	-1.64
MetaReg - Balaji et al. (2018)	79.9	75.1	95.2	69.5	79.9	81.7	+2.25
MLDG - Li et al. (2018a)	79.5	77.3	94.3	71.5	79.1	80.7	+2.02
Epi-FCD - Li et al. (2019)	**82.1**	77.0	93.9	73.0	79.1	81.5	+3.03
JiGen - Carlucci et al. (2019)	79.4	75.3	**96.0**	71.4	79.1	80.5	+1.77
MASF - Dou et al. (2019)	80.3	75.7	94.3	69.6	79.2	80.0	+1.01
DeepAll	75.8	73	94.4	70.9	-	78.5	-
BNE (*Ours*)	78.8	**78.9**	94.8	**79.7**	78.5	**83.1**	**+5.86**

shift. Each domain includes samples from 7 different categories, for a total of 9991 samples. Some examples are shown in Fig. 2. (b.) *Office-31* (Saenko et al. (2010)) was originally introduced for the task of domain adaptation and has been subsequently used for domain generalization as well. The dataset is composed of 3 different sources and 31 categories, representing images captured with a Webcam and a dSLR camera or collected from the Amazon website. In the supplementary material we report results also for the *Office-Caltech* (Gong et al. (2012)) dataset and expand the evaluation on PACS and Office-31.

Evaluation Protocol. Coherently with other works, we evaluate both the AlexNet (Krizhevsky et al. (2012)) and the more recent ResNet-18 (He et al. (2016)) architectures. For the experiments on PACS and Office-31 we follow the standard *leave-one-domain-out* evaluation procedure, where the model is trained on all domains but one and tested on the left-out one. However, since the original version of AlexNet does not include batch normalization layers, we adopt a variant with batch normalization applied on the activations of each convolutional layer (Simon et al. (2016)). Since the goal of domain generalization is to leverage multiple sources to train models that are robust on any target domain, the natural deep-learning baseline to compare against consists in training directly on the merged set of source domains. We will refer to it as (DeepAll). We compare our method against this strong baseline and to several deep-learning based state-of-the-art methods for domain generalization. Since different methods rely on different initialization of network weights which result in different baselines, we compare to methods providing their own baseline and report for every competitor: the performance on each unseen domain, the average baseline performance (Avg. DA), the average performance of the method itself (Avg.) and the relative gain (Δ%).

4.2 Domain Generalization for Classification

We compare models for the task of object classification on commonly used benchmarks.

4.2.1 PACS

We first benchmark our method on the PACS dataset (Li et al. (2017)), which presents a challenging domain generalization setting for object recognition. Every test uses 3 domains as training set and one as unknown test set; for each of this leave-one-out configuration we train a model from the same initialization for 60 epochs. We first test our method using the ResNet-18 architecture and report the results in Tab. 1. Overall, our proposal obtains the best absolute accuracy across on 2 out of 4 target sets, with an average accuracy (Avg.) of 83.1% and a relative gain (Δ%) almost double than the closest competitor. Since all the networks are initialized with weights trained on ImageNet they are implicitly biased towards the *Photo* domain, as testified by the higher accuracy on it when treated as test set. *Sketch*, instead, is arguably the more challenging and different from ImageNet as testified by the lower accuracy achieved by all methods and it is when testing on it that our method provides the bigger gain (+9.6% absolute gain in accuracy over our baseline and +8.4% over the strongest competitor).

4.2.2 Office-31

We evaluate our method on Office-31 (Saenko et al. (2010)) according to the *leave-one-domain-out* protocol. We use AlexNet initialized with ImageNet weights to compare with published results and train it with our method for 100 epochs with learning rate 10^{-4}. Tab. 2 shows that our approach

Table 2: State-of-the-art comparison on Office-31 with AlexNet.

Method	Amazon	Dslr	Webcam	Avg. DA	Avg.	Δ%
UB - Khosla et al. (2012)	42.4	98.5	93.4	74.2	78.1	+5.26
DSN - Bousmalis et al. (2016)	44.0	99.0	94.5	74.2	79.2	+6.74
MTAE - Ghifary et al. (2015)	43.7	99.0	94.2	74.2	79.0	+6.47
DGLRC - Ding & Fu (2017)	45.4	**99.4**	**95.3**	74.2	80.0	+7.82
MCIT - Rahman et al. (2019)	51.7	97.9	94.0	74.2	81.2	**+9.43**
DeepAll	43.8	94.1	88.4	-	75.4	-
BNE (*Ours*)	**54.0**	**99.4**	92.3	75.4	**81.9**	+8.62

Table 3: Comparison of different variants of our method on PACS with Resnet.

Method	DT	Warm-up	Art	Cartoon	Photo	Sketch	Avg.	Δ%
DeepAll	-	-	75.8	73.0	94.4	70.9	78.5	-
(a) BNE	✗	✗	74.7	71.7	93.0	74.8	78.6	+0.03
(b) BNE	✓	✗	**79.8**	76.0	92.5	72.5	80.2	+2.13
(c) BNE	✓	✓	78.8	**78.9**	**94.8**	**79.7**	**83.1**	**+5.86**
(d) DNet	✓	✓	77.3	73.8	94.2	71.2	79.1	+0.08

obtains the best absolute accuracy in two out of three test scenarios and a relative gain comparable or better than the alternatives. The *Amazon* target domain is the more challenging as the images are acquired in ideal conditions (*i.e.,* white backgrounds, studio lighting...) that are fairly different from the ImageNet domain. On it, we boost the absolute accuracy with respect to the baseline by an impressive $+11.2\%$ and $+2.3\%$ with respect to the closest competitor.

4.3 Ablation Study

We want to measure the impact on performance of the different components of our methods. We run ablation experiments on the PACS dataset using the ResNet-18 architecture and report the results in Tab. 3, comparing again with the DeepAll baseline and to an alternative method (DNet, short for '*DiscoveryNet*' on row (d)) inspired by Mancini et al. (2018b). In the latter domain membership is assigned by a domain discovery network instead of computing the distance between the latent batch normalization embeddings as we propose. More details on the domain discovery network are reported in the supplementary material. On row (a) we show the performance gained by using separate batchnorm statistics for the different train domains and using the projection and weighting strategy described in Sec. 3.3; row (b) extends the method above by using the *distance weighting at training time* (DT) as described in Sec. 3.4; finally, row (c) includes a warm-up phase in the training of the model to make population statistics converge to stable values before starting the distance training. By comparing the average accuracy (Avg.) across the four possible target sets, it is clear how every component contributes to an increase in performance with respect to the baseline. Comparing line (c) to (d) we can see how our proposal is more effective and require less parameters than the alternative domain mapping strategy inspired by Mancini et al. (2018b).

5 Conclusions

Our method allows to navigate in the latent space of batch normalization statistics to describe unknown domains as a combination of the known ones. We rely on domain-specific normalization layers to keep different representations for the different domains, then use this implicit representations to localize samples from unknown domains. We explicitly keep separate per domain representations while previous works mainly focused on deriving domain-invariant ones. We believe that this work highlights some not yet well explored properties of the batch normalization layers. Our formulation could also be easily adapted to the domain adaptation setting by injecting unlabelled samples from the target domain during training. In the case where few target samples are available at the same time, they could all be used to produce a less biased estimate of the unseen domain statistics. Those are both promising directions that we plan to explore in future works.

REFERENCES

Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. Metareg: Towards domain generalization using meta-regularization. In *Advances in Neural Information Processing Systems*, pp. 998–1008, 2018.

Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. Domain separation networks. In *Advances in neural information processing systems*, pp. 343–351, 2016.

John S Bridle. Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition. In *Neurocomputing*, pp. 227–236. Springer, 1990.

Fabio M Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2229–2238, 2019.

Fabio Maria Carlucci, Lorenzo Porzi, Barbara Caputo, Elisa Ricci, and Samuel Rota Bulo. Autodial: Automatic domain alignment layers. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 5077–5085. IEEE, 2017a.

Fabio Maria Carlucci, Lorenzo Porzi, Barbara Caputo, Elisa Ricci, and Samuel Rota Bulo. Just dial: Domain alignment layers for unsupervised domain adaptation. In *International Conference on Image Analysis and Processing*, pp. 357–369. Springer, 2017b.

Zhengming Ding and Yun Fu. Deep domain generalization with structured low-rank constraint. *IEEE Transactions on Image Processing*, 27(1):304–313, 2017.

Qi Dou, Daniel Coelho de Castro, Konstantinos Kamnitsas, and Ben Glocker. Domain generalization via model-agnostic learning of semantic features. In *Advances in Neural Information Processing Systems*, pp. 6447–6458, 2019.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1126–1135. JMLR. org, 2017.

Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. Domain generalization for object recognition with multi-task autoencoders. In *Proceedings of the IEEE international conference on computer vision*, pp. 2551–2559, 2015.

Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2066–2073. IEEE, 2012.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Zeyi Huang, Haohan Wang, Eric P Xing, and Dong Huang. Self-challenging improves cross-domain generalization. *arXiv preprint arXiv:2007.02454*, 2020.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pp. 448–456, 2015.

Aditya Khosla, Tinghui Zhou, Tomasz Malisiewicz, Alexei A Efros, and Antonio Torralba. Undoing the damage of dataset bias. In *European Conference on Computer Vision*, pp. 158–171. Springer, 2012.

Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2. Lille, 2015.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pp. 5542–5550, 2017.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018a.

Da Li, Jianshu Zhang, Yongxin Yang, Cong Liu, Yi-Zhe Song, and Timothy M Hospedales. Episodic training for domain generalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1446–1455, 2019.

Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5400–5409, 2018b.

Ya Li, Xinmei Tian, Mingming Gong, Yajing Liu, Tongliang Liu, Kun Zhang, and Dacheng Tao. Deep domain generalization via conditional invariant adversarial networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 624–639, 2018c.

Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*, 2016.

Antonio Loquercio, Elia Kaufmann, René Ranftl, Alexey Dosovitskiy, Vladlen Koltun, and Davide Scaramuzza. Deep drone racing: From simulation to reality with domain randomization. *IEEE Transactions on Robotics*, 2019.

Yawei Luo, Liang Zheng, Tao Guan, Junqing Yu, and Yi Yang. Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2507–2516, 2019.

Massimiliano Mancini, Samuel Rota Bulò, Barbara Caputo, and Elisa Ricci. Best sources forward: domain generalization through source-specific nets. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pp. 1353–1357. IEEE, 2018a.

Massimiliano Mancini, Samuel Rota Bulo, Barbara Caputo, and Elisa Ricci. Robust place categorization with deep domain generalization. *IEEE Robotics and Automation Letters*, 3(3):2093–2100, 2018b.

Massimiliano Mancini, Lorenzo Porzi, Samuel Rota Bulò, Barbara Caputo, and Elisa Ricci. Boosting domain adaptation by discovering latent domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3771–3780, 2018c.

Massimiliano Mancini, Samuel Rota Bulo, Barbara Caputo, and Elisa Ricci. Adagraph: Unifying predictive and continuous domain adaptation through graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6568–6577, 2019.

Saeid Motiian, Marco Piccirilli, Donald A Adjeroh, and Gianfranco Doretto. Unified deep supervised domain adaptation and generalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5715–5725, 2017.

Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *International Conference on Machine Learning*, pp. 10–18, 2013.

Mohammad Mahfujur Rahman, Clinton Fookes, Mahsa Baktashmotlagh, and Sridha Sridharan. Multi-component image translation for deep domain generalization. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 579–588. IEEE, 2019.

Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *European conference on computer vision*, pp. 213–226. Springer, 2010.

Shiv Shankar, Vihari Piratla, Soumen Chakrabarti, Siddhartha Chaudhuri, Preethi Jyothi, and Sunita Sarawagi. Generalizing across domains via cross-gradient training. *arXiv preprint arXiv:1804.10745*, 2018.

Marcel Simon, Erik Rodner, and Joachim Denzler. Imagenet pre-trained models with batch normalization. *arXiv preprint arXiv:1612.01452*, 2016.

Masashi Sugiyama and Amos J Storkey. Mixture regression for covariate shift. In *Advances in Neural Information Processing Systems*, pp. 1337–1344, 2007.

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 23–30. IEEE, 2017.

Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John C Duchi, Vittorio Murino, and Silvio Savarese. Generalizing to unseen domains via adversarial data augmentation. In *Advances in Neural Information Processing Systems*, pp. 5334–5344, 2018.

Mei Wang and Weihong Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312: 135–153, 2018.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pp. 3320–3328, 2014.