Mixing Up Domain Generalization with *Deep Domain MixUp*

Trenton Chang
Department of Computer Science
Stanford University
tchang97@stanford.edu

Bin (Claire) Zhang
Department of Electrical Engineering
Stanford University
zhangbin@stanford.edu

Abstract

Convolutional neural networks (CNNs) obtain impressive performance on a variety of computer vision tasks, but significant obstacles impede the real-world deployment of CNN-powered machine learning systems. One such desiderata is domain generalization, the ability to perform well on distributionally different data points. Previous domain generalization techniques are often intensive or difficult to train. We introduce Deep Domain MixUp (DDM), a plug-and-play supervision source which exploits naturally-occurring domain metadata to sample convex combinations of extracted features, constructing a richer estimate of the population feature distribution, which we hypothesize benefits out-of-domain generalization. We find that this method achieves high accuracy on two distributional shift benchmarks (93.2%, CAMELYON17 and 77.3%, iWILDCAM). Further ablations and visualization reveal that DDM improves domain indistinguishability, and that the DDM mixing distribution induces a tradeoff between in-domain and out-of-domain generalization. We hope that future work can integrate statistical insights about domain generalization to formalize the role of DDM in domain generalization.

1. Introduction

Convolutional neural networks (CNN) have achieved groundbreaking performance on image classification tasks and with transferrability across various problems. However, deploying machine learning models remains non-trivial in the real world. One obstacle is covariate shift, in which the testing data is distributionally different from the training data, a particular concern for low-data regimes that do not distributionally resemble the training data. Recent work has highlighted spurious correlations learned by CNNs in medical imagery or visual differences between images taken at different hospitals [30, 10], revealing the brittleness of models on out-of-distribution data.

Previous approaches to tackle domain generalization include maximizing cross-domain similarity by reweighting [8, 42], optimal transport [16, 51], adversarial training [22, 26, 33, 34, 45, 49, 24], or reconstruction-based methods [23, 52]. However, these approaches can be computationally intensive or challenging to train, making their practical usage difficult.

We aim to exploit naturally-occurring domain annotations in metadata to provide an easily trainable, computationally cheap scheme for domain generalization. Metadata induces a prior on the types of distributional shift that may emerge. For example, in a wildlife classification setting [3], one can expect to know which camera produced which image; the idea is to encode knowledge of "camera-level differences" implicitly to allow generalization to out-of-domain, or "new-camera" images. Current approaches implicitly leverage this knowledge, learning robust domain representations through auxilliary domain classification, source-to-target mappings, or as a pretraining task [50].

We propose *Deep Domain MixUp (DDM)*, an auxilliary task that trains a domain classification head with a gradient reversal layer using random convex combinations of extracted features. This builds directly on domain-adversarial training [22] and MixUp augmentation [53]. As deep layers of the network are hypothesized to have higher source-target distribution overlap, *DDM* is a natural technique for robustifying domain-adversarial methods. A general schematic of this method can be found in Fig. 1.

First, we find that domain adversarial training has high accuracy on two distributional shift benchmarks [32]. We find that *DDM* is effective in low-domain settings, serving as a regularizer, but is less effective in settings with large numbers of domains, due to the increased difficulty of domain generalization in that setting. Further analysis of the mixing parameter provides evidence that smaller mixing values are more effective for out-of-domain generalization at the expense of in-domain generalization. Qualitative analysis of learned domain-invariant representations under this paradigm further demonstrates *DDM*'s ability to reduce domain representation distinguishability.

The input to our algorithm are images with class labels

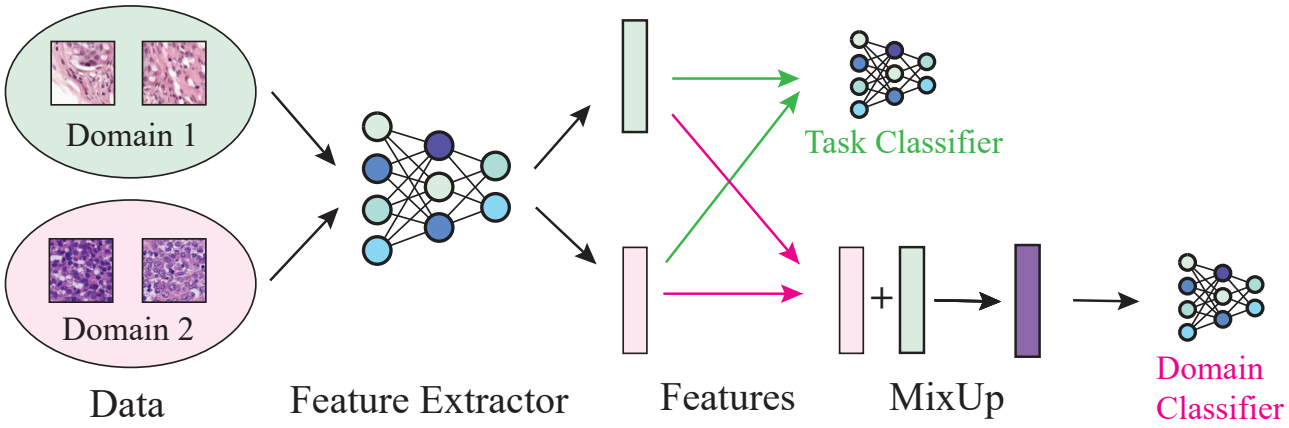


Figure 1: Schematic of *Deep Domain MixUp (DDM)* in the domain-adaptive machine learning pipeline. Features are extracted from images and stochastically mixed across domains for adversarial domain classification.

and domain labels. We use our proposed model to output predicted class labels, which are evaluated on. Domain labels and predictions are used exclusively during training to maximize domain confusion.

The remainder of our paper proceeds as follows. We discuss related work in Section 2. We overview the problem setup and our approach in Section 3. Then, we discuss the results of domain-adversarial training with *DDM* in Section 4. A brief discussion of ethical implications follows in Section 5, with concluding remarks in Section 6. Our code is available at `https://github.com/tchainzzz/spicy-dann` to facilitate reproducible research.

2. Related Work

Neural domain adaptation. A mathematical theory of domain adaptation was established in [5], which introduces $\mathcal{H}\Delta\mathcal{H}$-divergence as a metric measuring the distance between hypothesis classes. A key idea of successful domain adaptation is low $\mathcal{H}\Delta\mathcal{H}$, which translates to domain indistinguishability—a construct that subsequent neural domain adaptation methods attempt to model implicitly. Generative-adversarial methods [22, 26, 33, 34, 45, 49, 24] and reconstruction-based methods [23, 52] have shown to be successful in this regime. However, these incur significant computational cost; in contrast, our method attempts injects an auxilliary task that replaces supervision from reconstruction/generation.

Domain alignment. An important line of work in deep learning-based methods is domain alignment, which aims to learn domain-invariant representations and allow models to have high generalization ability. This is commonly achieved by statistically closing the distance between feature distributions of different domains [46, 35, 44, 43, 54], with Maximum Mean Discrepancy (MMD) as a widely used domain similarity measure example [46]. Another popular approach is to adversarially force feature representations to become indistinguishable [22, 21, 15, 11, 38], with DANN being one exceptional work [22, 21] for its state-of-the-art domain adaptation performance, despite high computational cost. In essence, DANN trains a domain discriminator to distinguish source and target domains, and another feature extractor to fool the discriminator. We want to expand on DANN for its compatibility with our feature mixup approach and great potential in domain generalization.

Train-time data/label modification. MixUp in its original form [53] can be seen as an extension of data augmentation, a method of specifying label-preserving input transformations. Data augmentation in computer vision has been extensively explored: geometric, color-space, and cropping transformations are commonplace in the standard repertoire of augmentations [13, 14, 19, 29, 55]. GAN and style-transfer based approaches seek to directly generate synthetic examples [7, 40, 2, 36, 20, 37, 56]. Other methods smooth labels [39] or apply transformations to extracted features more similarly to our work [18]. While pixel-space transformations are useful when there are explicit visual invariants, in contrast, we manipulate the extracted features directly based on the assumption that deep layers learn domain-invariant features.

3. Methods

In this section, we provide preliminaries about the problem and our approach. We first discuss domain general-

ization in Subsection 3.1, the task of training models to perform well in settings with high covariate shift. We explain domain-adversarial training in Subsection 3.2, which we use in our experiments. Then, we present our method: *DDM*, an auxilliary task for domain-adversarial training, in Subsection 3.3. We overview the datasets and models used in Subsection 3.4.

3.1. Problem Setup: Domain Generalization

We consider a C-way classification task. The model works with input image samples $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, ...\mathbf{x}_n\} \in X$, where X is the input space, and class labels $\mathrm{y} = \{\mathrm{y}_1, \mathrm{y}_2, ...\mathrm{y}_n\}$ from the output space Y, where Y is a categorical label ($Y \in \{1, 2, ...C\}$). We assume there exists two data-generating distributions $\mathcal{S}$ and $\mathcal{T}$ exhibiting covariate shift.

We can model domain-informative metadata by decomposing $\mathcal{S}$ and $\mathcal{T}$ exactly into their constituent domains. We can write that for $\{\mathcal{D}_1, \ldots, \mathcal{D}_k\}, \mathcal{S} = \cup_{i=1}^{k} \mathcal{D}_i$, where $\mathcal{D}_i \cap \mathcal{D}_j = \emptyset$ for all $i \neq j$ (all domains mutually disjoint), and likewise for k' testing domains in $\mathcal{T}$. Practically, for an image dataset, the domains $\mathcal{D}_i$ might correspond to task-dependent variation like different camera angles or different locations, among other potential variations.

We proceed with a variation of the standard ERM setup. During training time, the model draws training examples $(\mathbf{x}, \mathbf{y}, \mathbf{d}) \overset{i.i.d}{\sim} \mathcal{S}$ for example $\mathbf{x}$, label y, and domain indicator $\mathbf{d}$. Then, marginalizing over the set of domains, we minimize the objective function

$$\mathbb{E}_{\mathcal{D}_i \sim \mathcal{S}}[\mathbb{E}_{(\mathbf{x}_i, \mathbf{y}_i) \sim \mathcal{D}_i}[\ell(\mathbf{y}_i, f_c(f_e(\mathbf{x}_i)))]] \tag{1}$$

where f_e is the feature extractor and f_c is the classifier. Let $f^*_{\mathcal{S}}$ be the optimal classifier; i.e. $(f^*_c \circ f^*_e)$ as trained on $\mathcal{S}$ with domain annotations. Under the domain generalization setup, the model trained on data from $\mathcal{S}$ is evaluated on $(\mathbf{x}, \mathbf{y}) \overset{i.i.d}{\sim} \mathcal{T}$. We hope for

$$\mathop{\mathbb{E}}_{(\mathbf{x}_i, \mathbf{y}_i) \sim \mathcal{T}}[\ell(\mathbf{y}_i, f^*_{\mathcal{S}}(\mathbf{x}_i))] - \mathop{\mathbb{E}}_{\mathcal{D}_i \sim [k]} \left[\mathop{\mathbb{E}}_{(\mathbf{x}_i, \mathbf{y}_i) \sim \mathcal{D}_i} [\ell(\mathbf{y}_i, f^*_{\mathcal{S}}(\mathbf{x}_i)] \right], \tag{2}$$

or the gap between test accuracy and training accuracy, to be small, which corresponds to good out-of-domain performance. Notably, this differs from the standard ERM setup in that we do not assume that $\mathcal{S}, \mathcal{T}$ are identically distributed.

3.2. Neural Domain Adaptation with Domain-Adversarial Learning

We now detail our implementation of a domain-adversarial neural network. Neural techniques for domain-invariance based domain generalization include Domain-Adversarial Neural Networks (DANN) [22] or Domain Separation Networks (DSN) [6], which learn domain-invariant

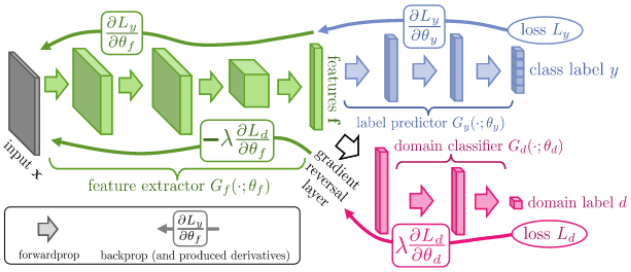


Figure 2: Schematic of a domain-adversarial neural network (DANN), which our method is based on, taken from [22].

representations. At a high-level, this is done by performing two classification tasks: task classification and domain classification. Task classification simply refers to the primary task an architecture is designed to solve; domain classification pertains to identifying the domain that originated a particular example. Intuitively, for a domain-invariant representation, we want to penalize correct domain classifications while rewarding correct task classifications.

For simplicity, we build our approach on top of DANN, although our method can be applied to any technique with a domain-classification/representation component. DANN learns domain representations in order to maximize domain confusion, or maximize loss on the domain classification task. DANN achieves this using a convolutional feature extractor, and then passing the resultant representation into two heads corresponding to each task. Both heads are fully-connected neural networks for classification, which can be trained with gradient descent on standard cross-entropy loss. Notably, the domain-classification head features a *gradient reversal layer*, which results in gradient descent-based optimization on task classification loss with simultaneous gradient ascent-based optimization on domain classification loss.

Formally, for feature extractor f_e, domain classification head f_d, and task classifier f_c with corresponding parameters $\theta_e, \theta_d, \theta_c$, define $z_c^{(i)} = f_c(f_e(\mathbf{x}^{(i)}))$ to be the task class prediction for example $\mathbf{x}^{(i)}$ with target $\mathrm{y}^{(i)}$, and $z_d^{(i)} = f_d(f_e(x^{(i)}))$ for the domain prediction with target $d^{(i)}$. We take ℓ_c and ℓ_d as task classification and domain classification losses. Then we solve

$$\min_{\theta_c, \theta_e} \max_{\theta_d} \mathop{\mathbb{E}}_{(x,y,d) \sim \mathcal{S}} [\ell_c(z_c, y) - \rho \ell_c(z_d, d)] \tag{3}$$

where ρ is the weight hyperparameter controlling the importance of the domain classification task. This shows that domain-adversarial training uses domain confusion as a regularizer, which is rigorously explored in more detail in Appendix A. This differs from the standard DANN objective in that we assume no access to unlabeled test data; mathe-

matical implications of this are explored in the **Appendix A**. In our experiments, we set $\rho = 1$.

3.3. Deep Domain MixUp

Domain-adversarial methods induce domain invariance by penalizing domain-discriminative features through an adversarial objective. We robustify the domain classification task by introducing the *DDM* auxilliary task, a special case of the MixUp method to extracted features at deeper layers in the model. The motivation for MixUp comes from vicinal learning: creating perturbed training examples by taking convex combinations biases the model to preserve linearity in the input space at deep layers of the network [53]. This empirically improves adversarial robustness and stabilizes GAN training. Formally, to perform MixUp, we sample two data points $\mathbf{x}_i, \mathbf{x}_j$ with corresponding domain labels z_i, z_j. We generate a synthetic example-label pair $\mathbf{x}', z'$ by sampling $\lambda \sim \text{Beta}(\alpha, \alpha)$ and computing

$$\mathbf{x}' = \lambda \mathbf{x}_i + (1 - \lambda)\mathbf{x}_j \tag{4}$$

$$z' = \lambda z_i + (1 - \lambda) z_j. \tag{5}$$

where α is a hyperparameter.

However, applying this technique to the pixel-space may be insufficient for domain generalization for two reasons: 1) low source-target domain overlap, and 2) the noisiness of pixel-space metrics. A core principle of domain generalization is that the source and target distribution should have similar supports, whether enforced by learning a mapping between distributions, as in optimal transport [12], or a domain-invariant representation as in $\mathcal{H}\Delta\mathcal{H}$-divergence theory [4]. Furthermore, not only are pixel-space metrics often uninformative for images, but also, pixel-space MixUp may preserve low domain overlap. Concretely, pixel-space MixUp generates synthetic data points that remain disjoint from the target distribution support in cases with low source-target domain overlap, which is often the assumption in domain generalization.

This is especially suitable for domain classification tasks, as the number of domains is generally low, such that our estimate of the domain-invariant distribution support may be poorly conditioned or otherwise high-variance. Compellingly, this is a model- and task-agnostic auxilliary supervision source applicable to any neural domain adaptation approach that model domain invariance. Furthermore, this method attains minimal computational overhead as compared to generative/GAN-based techniques like CycleGAN [56] and Cycada [26].

3.4. Datasets and Models

iWILCAM2020-WILDS. `iWILDCAM2020-WILDS` is a variant of the iWILDCAM2020 dataset [3], which consists of 203,029 RGB wildlife images from 323 camera traps, featuring 181 animal species and an "empty" class. This data is useful for biodiversity studies. The images feature varying illumination, angle, background, and other characteristics. Machine learning models trained on images from certain cameras may generalize poorly on images from other cameras. Here, each camera trap is considered a domain. All images are resized to 448x448 pixels. No further preprocessing is applied.

CAMELYON17-WILDS. `CAMELYON17-WILDS` is a variant of the Camelyon17 dataset [9], consisting of 450,000 RGB 96x96 histopathological image patches from 50 whole-slide images for tumor tissue studies in 5 hospitals. Hospitals use different data collection and processing pipelines, i.e. immunostaining, imaging steps, or machinery differences, such that images from different hospitals may look different, hindering the deployment of machine learning in this setting. Here, each hospital is considered a domain. This is a binary classification task for the presence of tumor tissue in the central 32x32 region of the image patch. We use the original 96x96 sizing in our experiments. No further preprocessing is applied.

Modeling and evaluation. For the `iWILDCAM2020-WILDS` benchmark, we use a ResNet-50 [25] backbone; for `CAMELYON17-WILDS`, we use a DenseNet121 [28] backbone. The domain classification head consists of a three-layer fully-connected neural net with 1024, 1024, and D neurons, where D is the number of domains in a task. Our hyperparameters are taken directly from [32] as opposed to manual tuning to enable fair comparison. We train the DenseNet121 using stochastic gradient descent with a learning rate of 10^{-3} and momentum 0.9, with weight-decay regularization with weight 0.01; the ResNet-50 uses the Adam [31] optimizer with learning rate 3×10^{-5}. Since domain-adversarial training has yet to be validated on WILDS datasets to the best of our knowledge, as a baseline, we first validate domain-adversarial training *without DDM*. We then evaluate our method based on out-of-domain (OOD) performance and worst-case domain accuracy, and provide figures on an official in-domain validation split as well. We compare to ERM, CORAL, IRM, and Group DRO as domain generalization baselines, as reported in [32].

4. Results and Discussion

In this section, we discuss the results of domain-adversarial training on domain generalization benchmarks with *DDM*. In Section 4.1, we discuss performance on `Camelyon17-WILDS`, highlighting the success of *DDM* in a low-domain setting. In Section 4.2, we discuss per-

formance on `iWILCAM2020-WILDS`, highlighting the viability of domain-adversarial training in high-domain settings; however, the performance of *DDM* is less impressive in high-domain settings. To understand the dynamics of *DDM*, we run an ablation study on the mixing parameter distribution in Section 4.3, finding a tradeoff between out-of-domain and in-domain accuracy as we vary the mixing parameter, with conservative mixtures being preferred. Lastly, in Section 4.4, a qualitative analysis of domain-invariant representations via t-SNE reveals reduced domain representation distinguishability by *DDM*.

4.1. Results on Camelyon17-WILDS

We train a DenseNet121 from scratch with DANN, which we call DenseDANN. We follow the setup of WILDS [32] except for batch size; which we set to 96, resulting in 32 examples from each of the three training domains. We stop training after 3 epochs because the network memorizes the training set. For comparison, we include numbers from WILDS baselines.

DenseDANN beats all other baselines, exceeding ERM by 6.6%, and the best domain generalization methods (CORAL and IRM) by 5.3%. This shows that domain-adversarial training provides significant lift for domain generalization. However, there remains a significant gap between in-distribution validation accuracy and out-of-distribution test accuracy of 7.4% on the no-MixUp setup, and of 5.8% on the MixUp setup. [32] reports gaps of magnitude for the WILDS baselines with respect to in-distribution validation accuracy.

Adding *DDM* as an auxilliary task stabilizes the convergence of the domain classification head of DenseDANN. To see this, note that the oscillations in the domain classification objective have significantly smaller amplitude for *DDM* (cyan) than the *DDM*-free run (yellow) in Fig. 3. before converging around $\log_2(3)$, as expected (recall that `Camelyon17-WILDS` features three training domains). It also improves accuracy, with a 1.7% lift over *DDM*-free

Method	Acc. (ID)	Acc. (OOD)
ERM [48]	93.2	84.9
CORAL [44]	95.4	86.2
IRM [1]	91.6	86.2
Group DRO [27, 42]	93.7	85.5
DenseDANN [22] (our impl.)	98.9	91.5
DenseDANN-DDM (ours)	99.0	93.2

Table 1: Validation accuracy of ERM, CORAL, IRM, Group DRO, and our DANN-based methods on Camelyon17-WILDS on an in-distribution validation set (ID) and an out-of-distribution (OOD) test set.

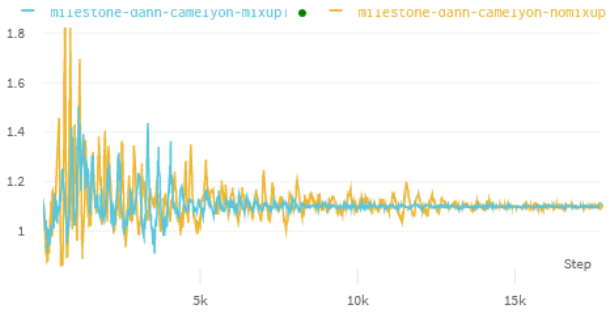


Figure 3: Domain classification loss without mixup (yellow) and with *DDM* (cyan). *DDM* stabilizes the domain classifier training significantly, enabling much faster convergence.

DANN at 93.2% out-of-domain validation accuracy, and functionally equal in-domain accuracy (lift of 0.1%). As with DenseDANN, incorporating *DDM* results in performance far superior to ERM and other domain generalization baselines, and draws close to the SOTA figure (94.8%) to the best of our knowledge. [41] These promising results highlight the potential utility of *DDM* as an appropriate auxilliary task for domain adaptation.

4.2. Results on iWILDCAM2020-WILDS

We train a ResNet50 initialized with Imagenet [17] weights with DANN applied, which we call ResDANN. We follow the setup of WILDS [32] with stratified sampling across domains. First, domain-adversarial training appears to be viable on this task, as ResDANN achieves a 14.6% lift over the best baseline (ERM) on out of domain data. We see a tradeoff with in-domain accuracy, however; the ResDANN models fail to exceed the performance of CORAL or even ERM.

Furthermore, the performance of *DDM* is somewhat lower in this setting. While *DDM* improves in-domain generalization, it fails to improve performance for out-of-domain generalization with respect to the ResDANN baseline, and in fact, even hurts performance by 3.9%. There are two possible explanations, both associated with the difficulty of constructing an unbiased, sufficiently dense estimate of the feature distribution. One is the increased dimensionality of the ResNet50 features (2048) as compared to the DenseNet121 features (1024), such that MixUp-based sampling cannot overcome the curse-of-dimensionality, leading to inherently sparse samples of the empirical feature distribution. The second is the large number of training domains in iWildCAM (243); the number of possible mixtures of domains scales quadratically with the number of domains, which again makes sufficiently

Method	Acc. (ID)	Acc. (OOD)
ERM [48]	82.5	62.7
CORAL [44]	81.8	60.3
IRM [1]	66.9	47.2
Group DRO [27, 42]	79.3	60.0
ResDANN [22] (our impl.)	80.0	77.3
ResDANN-DDM (ours)	80.8	73.4

Table 2: Accuracy of ERM, CORAL, IRM, Group DRO, and our DANN-based methods on iWILDCAM2020-WILDS on in-distribution (ID) validation and out-of-distribution (OOD) test data.

dense estimates of the feature distribution difficult. This makes domain-adversarial training more unstable due to the weakened supervision signal, ultimately impacting downstream performance. High levels of class imbalance in iWILDCAM2020-WILDS, as documented in [32], may further exacerbate this gap.[1]

Nevertheless, even with *DDM*, ResDANN outperforms all baselines by at least 10.7% with respect to out-of-distribution accuracy. Given the success of *DDM* on `Camelyon17-WILDS`, *DDM* remains a potentially viable training strategy. To better understand whether *DDM* is effective at all, immediate future work may seek to pinpoint whether its lack of success in `iWILDCAM2020-WILDS` is due to increased domain diversity, class imbalance, or an alternative cause.

4.3. Ablation Study: Understanding the MixUp Parameter

In our previous set of experiments, we draw the mixing parameter for MixUp from $Beta(1, 1)$, which is a uniform distribution. However, the effect of the mixing distribution on domain generalization is unclear. We investigate this empirically on Camelyon17, drawing the mixing parameters from $Beta(0.5, 0.5)$ and $Beta(2, 2)$. Intuitively, sampling $\lambda \sim Beta(0.5, 0.5)$ concentrates λ near 0 or 1, resulting in low expected mixing. Conversely, sampling $\lambda \sim Beta(2, 2)$ concentrates λ near 0.5, which results in more "mixed" examples.

First, we find that in-domain validation performance trades off with out-of-domain performance, as seen in Table 3. Specifically, as we increase MixUp parameter α, encouraging more evenly mixed samples, in-distribution validation accuracy improves, consistent with findings in the original MixUp work [53]. **At the same time, out-of-distribution accuracy decreases.** This is attributable to the domain-overlap theory of neural domain adaptation: to reiterate, we assume that at advanced layers of the network, a model successful at domain generalization will have learned a domain-invariant representation. If this domain-overlap assumption holds, increasing α concentrates the distribution of extracted features towards the analytic center of the convex hull of training domain representations. This decreases the diversity of augmented examples generated via MixUp.

While a bias-variance theory of domain adaptation has yet to be formalize to our knowledge, the concentration of the empirical distribution of extracted features can be interpreted as a low-variance, high-bias estimate of the population feature distribution. Thus, the drop in sample diversity can hurt out-of-domain generalization if the support of unseen domains lies far in feature-space to the seen training domains. However, this is unlikely to affect in-domain performance, as the center of the convex hull of training domain representations remains likely close in feature-space to unseen, in-domain points, which is empirically consistent with [53]. A geometric interpretation of this effect in 2D is shown in Fig. 4 (left).

Secondly, we make the surprising finding that **conservative combinations are more effective for improving out-of-domain performance**, that is, in the $\alpha < 1$ MixUp setting, where most sampled mixing parameters λ are close to 0 or 1. This suggests that the extrema of the convex hull of domain representations are a better estimate of out-of-domain data, as visualized (in 2D) in Fig. 4 (right). As the empirical distribution of extracted features is now more dispersed, we can interpret this setting as a high-variance estimate of the population feature distribution.

Our conclusions in our ablation study of α point to a need for a formal statistical understanding of domain generalization. In particular, first, we empirically observe a tradeoff between in-domain generalization and out-of-domain generalization. Then, we also find that conservative MixUp combinations are correlated with better out-of-domain performance. While the geometric explanation of Fig. 4 provides a simple, high-level intuition, to our knowledge, the statistical/analytical tools for formally understanding the role of MixUp in model generalization have not yet been developed.

MixUp Param. (α)	Acc. (ID)	Acc. (OOD)
0.5	95.8	93.0
1	99.0	90.0
2	98.5	88.0

Table 3: Ablation study on the role of MixUp parameter α (for sampling mixing parameter $\lambda \sim Beta(\alpha, \alpha)$, with $\alpha = \{0.5, 1, 2\}$.

[1] As an aside, we note that Macro F1 scores (not shown here) are quite low and even underperform the baselines in some situations, highlighting a potential challenge of domain-adversarial training in an imbalanced-class setting. However, this is beyond the scope of this work.

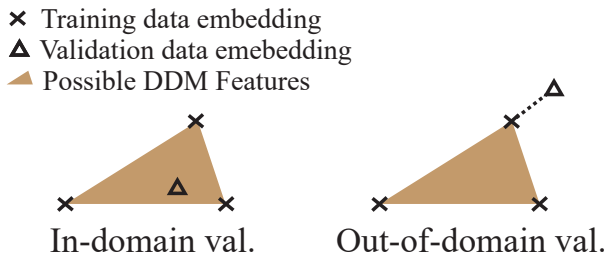


Figure 4: 2D geometric interpretation (not to scale) of how *Deep Domain MixUp (DDM)* interacts with in-domain (left) and out-of-domain (right) generalization. Results suggest that sparse/conservative mixtures can help out-of-domain generalization.

4.4. Qualitative Analysis of Domain-Invariant Representations

To better understand the representations extracted by each architecture with and without *DDM*, we use t-SNE [47] to visualize the feature embeddings in 2D. For each dataset, we construct a stratified sample of 1000 data points, sampling data points from the training, validation, and test sets with weights proportional to the number of domains in each split. As a denoising step, we apply principal component analysis (PCA) to reduce feature dimensionality such that the condition number of the feature covariance matrix is bounded above by 100 (Camelyon17-WILDS: 1024 $\rightarrow$ 40 features; iWILDCAM2020-WILDS: 2048 $\rightarrow$ 400 features). For t-SNE, we visualize using perplexity 50 and a learning rate of 20 over 5000 iterations, which qualitatively yielded the most stable and interpretable results. Training, validation, and test data points are represented by blue, green, and orange shades, respectively, with each single shade corresponding to one domain. Overall, as seen in Fig. 5, we observe that *DDM* help reduce domain distinguishability as features from each data split (train, val, or test) become less constrained in their local regime and less class discriminative after *DDM* is applied. Further analysis on each dataset are detailed below.

Results for the low-domain case of Camelyon17-WILDS are illustrated in Fig. 5(a) and 5(b), where only 3 train domains, 1 val domain, and 1 test domain are available. Distribution of baseline features in Fig. 5(a) demonstrate that features from each data split closely cluster in its own regime and each regime spans widely across the feature landscape, which are likely contributed by the low domain numbers and high feature dimension. The three feature regimes are similarly shaped and moderately overlapped with some observable difference. Adding *DDM* disrupts the local regime compactness, as shown in Fig. 5(b), as each regime span more space and features within each regime become more scattered. *DDM* also make all three regimes blend into one another more evenly. These observations suggest that *DDM* improves domain confusion both within each data split and across all data in the low-domain scenario, potentially contributing to the superior results in Table 1.

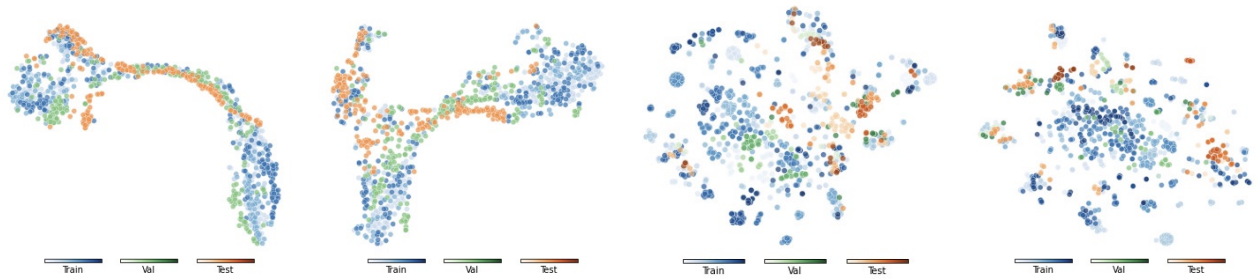
In the high-domain case of iWILDCAM2020-WILDS, as illustrated in Fig. 5(c) and 5(d), data points are sampled from a total of 243 train domains, 32 val domains, and 48 test domains. In Fig. 5(a), baseline features do not exhibit compact regimes due to the significantly larger domain volume. Features from the val and test splits, which have fewer domains than the train split, demonstrate stronger and visually separable localities. Adding *DDM* pulls in the train domain features while pushing out the val and test domain features towards the boundary of the train feature regime, as illustrated in Fig. 5(d). Such phenomenon echos the finding in Section 4.3, which states that boundary of the domain representations may better estimate out-of-domain data, but suggests that this finding may only apply in the high-domain scenario and not in the low-domain scenario, as similar phenomenon is not observed in Fig. 5(a) and 5(b). In addition, *DDM* does disrupt the localities of val and test feature regimes in the high-domain scenario, but does not blend three regimes as evenly as in the low-domain scenario, possibly due to the large gap in the total domain numbers. This observation potentially contributes to the slightly lower accuracy in Table 2, suggesting that large domain volume adds in difficulty for *DDM* to improve domain indistinguishability.

5. Limitations & Ethical Considerations

As *DDM* is computationally cheaper than generative/reconstruction-based methods for domain adaptation, the success of this method can potentially increase the accessibility of machine learning deployment. However, for safety-critical domains, more rigorous understanding of the dynamics of domain generalization is necessary for real-world deployment, as performance on one unseen domain does not guarantees equal results on a different unseen domain. Furthermore, this work assumes that domain differences embedded within the training data will also have similar structure to differences between unseen domains, which may not always hold for all tasks.

6. Conclusion

In this paper, we present *DDM*, an auxilliary task for domain-alignment based neural domain adaptation methods. In low-domain settings, it regularizes extracted features for downstream task classification, improving out-of-domain generalization. Further work is still necessary to stabilize the performance of *DDM* in high-domain/imbalanced-class settings. Additional study of the mixing parameter highlights that smaller mixing values are



(a) (b) (c) (d)

Figure 5: t-SNE on representations extracted by models for each dataset. Blue: training domains; green: validation domains; orange: test domains. (a) Camelyon17-WILDS w/o *DDM*; (b) Camelyon17-WILDS w/ *DDM*; (c) iWILDCAM2020-WILDS w/o *DDM*; (b) iWILDCAM2020-WILDS w/ *DDM*.

more effective for out-of-domain generalization, signalling that the support of extracted in-domain features have little overlap with that of out-of-domain features.

Acknowledgements and Contributions

We thank Michael Xie and Michael Zhang for their helpful discussions in deciding our approach for domain generalization. Trenton Chang developed the modeling and dataloading pipelines, adapting pre-existing code for DANN and *DDM* to this project. Claire Zhang built the training pipeline and the feature visualization pipeline. Experimental design and approaches were done by mutual agreement. Writing work was distributed equally. Code for *DDM* and DANN was adapted from `https://github.com/facebookresearch/mixup-cifar10` and `https://github.com/fungtion/DANN_py3`, respectively.

References

[1] M. Arjovsky, L. Bottou, I. Gulrajani, and D. Lopez-Paz. Invariant risk minimization, 2020.

[2] C. Baur, S. Albarqouni, and N. Navab. Melanogans: High resolution skin lesion synthesis with gans, 2018.

[3] S. Beery, E. Cole, and A. Gjoka. The iwildcam 2020 competition dataset, 2020.

[4] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan. A theory of learning from different domains. *Machine learning*, 79(1):151–175, 2010.

[5] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira. Analysis of representations for domain adaptation. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2007.

[6] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan. Domain separation networks, 2016.

[7] C. Bowles, L. Chen, R. Guerrero, P. Bentley, R. Gunn, A. Hammers, D. A. Dickie, M. V. Hernández, J. Wardlaw, and D. Rueckert. Gan augmentation: Augmenting training data using generative adversarial networks, 2018.

[8] J. Byrd and Z. C. Lipton. What is the effect of importance weighting in deep learning?, 2019.

[9] P. Bándi, O. Geessink, Q. Manson, M. Van Dijk, M. Balkenhol, M. Hermsen, B. Ehteshami Bejnordi, B. Lee, K. Paeng, A. Zhong, Q. Li, F. G. Zanjani, S. Zinger, K. Fukuta, D. Komura, V. Ovtcharov, S. Cheng, S. Zeng, J. Thagaard, A. B. Dahl, H. Lin, H. Chen, L. Jacobsson, M. Hedlund, M. Çetin, E. Halıcı, H. Jackson, R. Chen, F. Both, J. Franke, H. Küsters-Vandevelde, W. Vreuls, P. Bult, B. van Ginneken, J. van der Laak, and G. Litjens. From detection of individual metastases to classification of lymph node status at the patient level: The camelyon17 challenge. *IEEE Transactions on Medical Imaging*, 38(2):550–560, 2019.

[10] D. C. Castro, I. Walker, and B. Glocker. Causality matters in medical imaging. *Nature Communications*, 11(1), Jul 2020.

[11] M. Chen, S. Zhao, H. Liu, and D. Cai. Adversarial-learned loss for domain adaptation, 2020.

[12] N. Courty, R. Flamary, D. Tuia, and A. Rakotomamonjy. Optimal transport for domain adaptation, 2016.

[13] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le. Autoaugment: Learning augmentation policies from data, 2019.

[14] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le. Randaugment: Practical automated data augmentation with a reduced search space, 2019.

[15] S. Cui, S. Wang, J. Zhuo, C. Su, Q. Huang, and Q. Tian. Gradually vanishing bridge for adversarial domain adaptation, 2020.

[16] B. B. Damodaran, B. Kellenberger, R. Flamary, D. Tuia, and N. Courty. Deepjdot: Deep joint distribution optimal transport for unsupervised domain adaptation, 2018.

[17] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[18] T. DeVries and G. W. Taylor. Dataset augmentation in feature space, 2017.

[19] T. DeVries and G. W. Taylor. Improved regularization of convolutional neural networks with cutout, 2017.

[20] M. Frid-Adar, I. Diamant, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan. Gan-based synthetic medical image augmentation for increased cnn performance in liver lesion classification. *Neurocomputing*, 321:321–331, Dec 2018.

[21] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation, 2015.

[22] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks, 2016.

[23] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li. Deep reconstruction-classification networks for unsupervised domain adaptation, 2016.

[24] K. Goel, A. Gu, Y. Li, and C. Ré. Model patching: Closing the subgroup performance gap with data augmentation, 2020.

[25] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015.

[26] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. A. Efros, and T. Darrell. Cycada: Cycle-consistent adversarial domain adaptation, 2017.

[27] W. Hu, G. Niu, I. Sato, and M. Sugiyama. Does distributionally robust supervised learning give robust classifiers?, 2018.

[28] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger. Densely connected convolutional networks, 2018.

[29] H. Inoue. Data augmentation by pairing samples for images classification, 2018.

[30] S. Jabbour, D. Fouhey, E. Kazerooni, M. W. Sjoding, and J. Wiens. Deep learning applied to chest x-rays: Exploiting and preventing shortcuts, 2020.

[31] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2017.

[32] P. W. Koh, S. Sagawa, H. Marklund, S. M. Xie, M. Zhang, A. Balsubramani, W. Hu, M. Yasunaga, R. L. Phillips, I. Gao, T. Lee, E. David, I. Stavness, W. Guo, B. A. Earnshaw, I. S. Haque, S. Beery, J. Leskovec, A. Kundaje, E. Pierson, S. Levine, C. Finn, and P. Liang. Wilds: A benchmark of in-the-wild distribution shifts, 2021.

[33] H. Li, S. J. Pan, S. Wang, and A. C. Kot. Domain generalization with adversarial feature learning. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5400–5409, 2018.

[34] Y. Li, M. Gong, X. Tian, T. Liu, and D. Tao. Domain generalization via conditional invariant representation, 2018.

[35] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks, 2015.

[36] A. Madani, M. Moradi, A. Karargyris, and T. Syeda-Mahmood. Chest x-ray generation and data augmentation for cardiovascular abnormality classification. In E. D. Angelini and B. A. Landman, editors, *Medical Imaging 2018: Image Processing*, volume 10574, pages 415 – 420. International Society for Optics and Photonics, SPIE, 2018.

[37] A. Mikołajczyk and M. Grochowski. Data augmentation for improving deep learning in image classification problem. In *2018 International Interdisciplinary PhD Workshop (IIPhDW)*, pages 117–122, 2018.

[38] Z. Pei, Z. Cao, M. Long, and J. Wang. Multi-adversarial domain adaptation, 2018.

[39] G. Pereyra, G. Tucker, J. Chorowski, Ł. Kaiser, and G. Hinton. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*, 2017.

[40] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2016.

[41] A. Robey, G. J. Pappas, and H. Hassani. Model-based domain generalization. *ArXiv*, abs/2102.11436, 2021.

[42] S. Sagawa, P. W. Koh, T. B. Hashimoto, and P. Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization, 2020.

[43] B. Sun, J. Feng, and K. Saenko. Return of frustratingly easy domain adaptation, 2015.

[44] B. Sun and K. Saenko. Deep coral: Correlation alignment for deep domain adaptation, 2016.

[45] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation, 2017.

[46] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell. Deep domain confusion: Maximizing for domain invariance, 2014.

[47] L. van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008.

[48] V. Vapnik. Principles of risk minimization for learning theory. In *Advances in neural information processing systems*, pages 831–838, 1992.

[49] M. Wulfmeier, A. Bewley, and I. Posner. Incremental adversarial domain adaptation for continually changing environments, 2018.

[50] S. M. Xie, A. Kumar, R. Jones, F. Khani, T. Ma, and P. Liang. In-n-out: Pre-training and self-training using auxiliary information for out-of-distribution robustness, 2021.

[51] R. Xu, P. Liu, L. Wang, C. Chen, and J. Wang. Reliable weighted optimal transport for unsupervised domain adaptation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4393–4402, 2020.

[52] J. Yang, W. An, S. Wang, X. Zhu, C. Yan, and J. Huang. Label-driven reconstruction for domain adaptation in semantic segmentation, 2020.

[53] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization, 2018.

[54] Y. Zhang, H. Tang, K. Jia, and M. Tan. Domain-symmetric networks for adversarial domain adaptation, 2019.

[55] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang. Random erasing data augmentation, 2017.

[56] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks, 2020.

A. Appendix: Domain-Adversarial Training is an $\mathcal{H}$-divergence Regularizer

We focus on the general domain adpatation setting first, where this is an assumption of access to some unlabeled

test data. We will refer to DANN domain classification term $\max_{\theta_d} \mathbb{E}[\ell_c(z_d, d)]$ as the "regularizer," since our objective is of the form $\sum \ell(f(x), y) + \lambda \cdot R(f)$.

First, we reintroduce the concept of $\mathcal{H}$-divergence as a metric of domain distinguishability, which is defined for a hypothesis class over source and target distributions $\mathcal{S}, \mathcal{T}$ as

$$d_{\mathcal{H}}(\mathcal{S}, \mathcal{T}) = 2 \sup_{\theta} \left| \Pr_{x \sim \mathcal{S}} [f_\theta(x) = 1] - \Pr_{x \sim \mathcal{T}} [f_\theta(x) = 1] \right|. \tag{6}$$

The empirical estimate, as previously proven [5], for samples $S \sim \mathcal{S}, T \sim \mathcal{T}$ of cardinalities n, n' where $n + n' = N$ is given by

$$\hat{d}_{\mathcal{H}}(S, T) = 2 \left[1 - \min_{\theta} \frac{1}{n} \left(\sum_{i=1}^{n} \mathbf{1}[f_\theta(x) = 0] + \frac{1}{n'} \sum_{i=n+1}^{N} \mathbf{1}[f_\theta(x) = 1] \right) \right]. \tag{7}$$

However, the core assumption of domain generalization is a lack of unlabeled test data. This means our objective essentially treats S, T as equivalent. We can thus generalize this to the multi-domain classification setting over k domains $\mathfrak{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_k\}$ for domain generalization by writing

$$\hat{d}_{\mathcal{H}}(\mathfrak{D}) = 2[1 - \min_{\theta} Acc(f, \mathfrak{D})]. \tag{8}$$

where Acc is domain classification accuracy.

Comparing this with our DANN objective, we see that we can treat ℓ_c, our regularization function, as a proxy for negative-accuracy. We provide the simplification for showing the equivalency here, using R_s as shorthand for the source risk as in [22]:

$$\theta_c^* = \arg\min_{\theta_c} R_s + \lambda \hat{d}_h \tag{9}$$

$$= \arg\min_{\theta_c} R_s + \lambda(2 - 2 \min_{\theta_d} Acc(f, \mathfrak{D})) \tag{10}$$

$$= \arg\min_{\theta_c} R_s - \lambda \min_{\theta_d} Acc(f, \mathfrak{D}) \tag{11}$$

$$= \arg\min_{\theta_c} \max_{\theta_d} R_s - \lambda \ell_c \tag{12}$$

which recovers our DANN objective.

This proves that if ℓ_c is the 0-1 loss, then our regularizer is exactly equivalent to the empirical $\mathcal{H}$-divergence, but then ℓ_c is a non-strongly convex objective, making it unsuitable for regularization. Thus, our regularization can be interpreted as an approximation of $\hat{d}_{\mathcal{H}}(\mathfrak{D})$, as explained in [22], with λ as a tradeoff parameter between the source risk and risk across training domains.

Ultimately, the result is that this formulation guarantees the standard bounds on excess risk as seen in [5], but only with respect to subgroup performance, i.e. training domains. While this is promising for label shift settings, the domain generalization setting, which can be interpreted as learning under extreme covariate shift, does not directly benefit from this result. Notably, this formulation causes us to lose all guaranteed bounds of domain adaptation to test domains—intuitively, with no access to even unlabeled test data, constructing an estimate of the test distribution feels futile. Thus, the solvability of this problem thus hinges on leveraging inter-domain similarity structure: that robustness to training domain differences implies robustness to testing domain differences.

Part of this could potentially be justified by applying i.i.d. assumptions to domain samples, such that $\mathcal{D}_{train}, \mathcal{D}_{test} \overset{i.i.d}{\sim} \mathfrak{D}$. This assumption means that in expectation, domain differences within each non-trivial i.i.d. subset of $\mathfrak{D}$ should be similar. Additionally, this provides an intuitive justification for MixUp: by augmenting inputs into the domain classification network, we artificially introduce representations of domain differences into the domain-training set, bootstrapping a denser sample of $\mathfrak{D}$. Unfortunately, this is not theoretically principled, and a more rigorous framework of generalization with respect to domains or domain differences instead of data points is worth further study.