DIVA: DOMAIN INVARIANT VARIATIONAL AUTOENCODER

Anonymous authors
Paper under double-blind review

ABSTRACT

We consider the problem of domain generalization: how to learn representations given data from a set of domains that generalize to data from a previously unseen domain. We propose the domain invariant VAE (DIVA), a generative model that tackles this problem by learning three independent latent subspaces, one for the class, one for the domain and one for the object itself. In addition, we highlight that due to the generative nature of our model we can also incorporate unlabeled data from known or previously unseen domains. This property is highly desirable in fields like medical imaging where labeled data is scarce. We experimentally evaluate our model on the rotated MNIST benchmark where we qualitatively show that 1) the learned subspaces are indeed complementary to each other, 2) we improve upon recent works on this task and 3) incorporating unlabelled data can boost the performance even further.

1 INTRODUCTION

Deep neural networks (DNNs) led to major breakthroughs in a variety of areas like computer vision and natural language processing. Despite their big successes recent research shows that DNNs learn the bias present in the training data. As a result they are not invariant to cues that are irrelevant to the actual task (Azulay & Weiss, 2018). This leads to a dramatic performance decrease when tested on data from a different distribution with a different bias (Torralba & Efros, 2011).

In domain generalization our goal is to learn representations from a set of similar distributions, here called domains, that can be transferred to a previously unseen domain during test time. A common motivating application where domain generalization is crucial is medical imaging (Blanchard et al., 2011; Muandet et al., 2013). For instance, in digital histopathology a typical task is the classification of benign and malignant tissue. However, the preparation of a histopathology slide includes the staining and scanning of tissue which can greatly vary between hospitals. Moreover, a sample from a patient could be preserved in different conditions. As a result, each patient data could be treated as a separate domain (Lafarge et al., 2017; Ciompi et al., 2017). Another problem commonly encountered in medical imaging is class label scarcity. Annotating medical images is an extremely time consuming task that requires expert knowledge. However, obtaining domain labels is surprisingly cheap, since hospitals generally store information about the patient (e.g., age and sex) and the medical equipment (e.g., manufacturer and settings). Therefore, we are interested in extending the domain generalization framework to be able to deal with additional unlabeled data. We hypothesize that additional unlabeled data can lead to better domain generalization.

In this paper, we propose to tackle domain generalization via a new deep generative model which we refer to as the domain invariant variational autoencoder (DIVA). We extend the variational autoencoder (VAE) framework (Kingma & Welling, 2013) by introducing independent latent representations for an object (e.g. an image), a class label and a domain label. Such partitioning of the latent space will encourage and guide the model to disentangle these sources of variation. Furthermore, in order to further control this disentanglement, we propose to weigh all Kullback-Leibler divergence terms from the priors to the variational posteriors, similarly to (Higgins et al., 2017). Finally, by virtue of having a generative model we can also naturally handle the semi-supervised scenario, similarly to Kingma et al. (2014). We evaluate our model on a version of the MNIST dataset where each domain corresponds to a specific rotation angle of the digits.

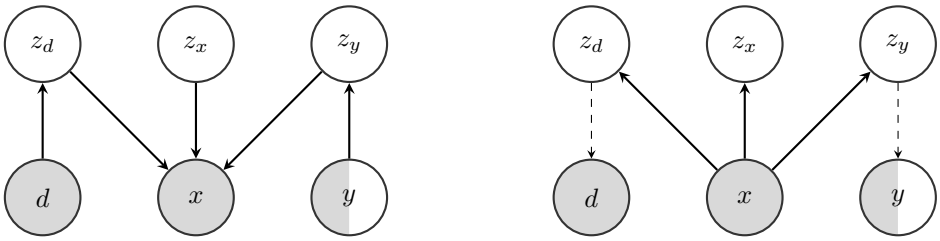

Figure 1: Left: Generative model. According to the graphical model we obtain $p(d, x, y, z_d, z_x, z_y) = p_\theta(x|z_d, z_x, z_y)p_{\theta_d}(z_d|d)p(z_x)p_{\theta_y}(z_y|y)p(d)p(y)$. Right: Inference model. We propose to factorize the variational posterior: $q_{\phi_d}(z_d|x)q_{\phi_x}(z_x|x)q_{\phi_y}(z_y|x)$. Dashed arrows represent auxiliary classifiers.

2 TOWARDS DOMAIN GENERALIZATION WITH GENERATIVE MODELS

2.1 DOMAIN GENERALIZATION

We follow the domain generalization definitions used in (Muandet et al., 2013). A domain is defined as a joint distribution $p(x, y)$ on $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ denotes an input space and $\mathcal{Y}$ an output space. Let $\mathfrak{P}_{\mathcal{X}\times\mathcal{Y}}$ be the set of all domains. We assume that we cannot observe domains directly. Instead, the training set consists of samples $\mathcal{S}$ taken from N domains, $\mathcal{S} = \{S^{(d=i)}\}_{i=1}^{N}$. The ith domain $p^{(d=i)}(x, y)$ is represented by n_i samples $S^{(d=i)} = \{(x_k^{(d=i)}, y_k^{(d=i)})\}_{k=1}^{n_i}$. Here each of the N distributions $p^{(d=1)}(x, y), \ldots, p^{(d=i)}(x, y), \ldots, p^{(d=N)}(x, y)$ is sampled from $\mathfrak{P}_{\mathcal{X}\times\mathcal{Y}}$. We further assume that $p^{(d=i)}(x, y) \neq p^{(d=j)}(x, y)$, therefore the samples in $\mathcal{S}$ are non i.i.d. During test time we are presented with samples $S^{(d=N+1)}$ from a previously unseen domain $p^{(d=N+1)}(x, y)$. We are interested in learning representations that generalize from $p^{(d=1)}(x, y), \ldots, p^{(d=N)}(x, y)$ to this new domain. Training data are given as tuples (d, x, y) in the case of supervised data or as (d, x) in case of unsupervised data.

2.2 DIVA: DOMAIN INVARIANT VAE

Assuming a perfectly disentangled latent space (Higgins et al., 2018), we hypothesize that there exists a latent subspace that is invariant to changes in d, i.e., that is domain invariant. We propose a generative model with three independent sources of variation; z_x, which is object specific, z_d, which is domain specific and finally z_y, which is class specific. While z_x keeps an independent Gaussian prior $p(z_x)$, z_d and z_y have conditional priors $p_{\theta_d}(z_d|d)$, $p_{\theta_y}(z_y|y)$ with learnable parameters θ_d, θ_y. This will encourage information about the domain d and label y to be encoded into z_d and z_y respectively. Furthermore, as z_d and z_y are marginally independent by construction, we argue that the model will learn representations z_y that are invariant with respect to the domain d. All three of these latent variables are then used by a single decoder $p_\theta(x|z_d, z_x, z_d)$ for the reconstruction of x.

Since we are interested in using neural networks for $p_\theta(x|z_d, z_x, z_d)$ exact inference will be intractable. For this reason, we perform amortized variational inference with an inference network (Kingma & Welling, 2013), i.e., we employ a variational autoencoder (VAE) framework. We introduce three separate encoders $q_{\phi_d}(z_d|x)$, $q_{\phi_x}(z_x|x)$ and $q_{\phi_y}(z_y|x)$ that serve as variational posteriors over these latent variables. Notice that we do not share their parameters as we empirically found that sharing parameters leads to a decreased generalization performance. For the prior and variational posterior distributions over the latent variables z_x, z_d, z_y we assume fully factorized Gaussians with parameters given as a function of their input. We coin the term domain invariant VAE (DIVA) for our overall model, which can be seen in Figure 1.

Given a specific dataset, all of the aforementioned parameters can then be optimized by maximizing the following variational lower bound

$$\begin{aligned}\mathcal{L}_s(d,x,y) = \mathbb{E}_{q_{\phi_d}(z_d|x)q_{\phi_x}(z_x|x),q_{\phi_y}(z_y|x)}&\left[\log p_\theta(x|z_d,z_x,z_y)\right] \\ &- \beta_d KL\left(q_{\phi_d}(z_d|x)||p_{\theta_d}(z_d|d)\right) - \beta_x KL\left(q_{\phi_x}(z_x|x)||p(z_x)\right) \\ &- \beta_y KL\left(q_{\phi_y}(z_y|x)||p_{\theta_y}(z_y|y)\right). \end{aligned} \tag{1}$$

Notice that we have introduced three separate weighting terms, β_d, β_x and β_y. These are motivated by the β-VAE (Higgins et al., 2017) and serve as a constraint that controls the capacity of the latent spaces of DIVA. Larger values of each β limit the capacity of each z and in the ideal case each dimension of z captures one of the conditionally independent factors in data. The β-VAE framework offers a trade-off between the information preservation, i.e., how well one can reconstruct x from the zs, and the capacity, i.e., how well the zs compresses information about x (Chen et al., 2018).

In order to further encourage separation of z_d and z_y into domain and class specific information respectively, we add two auxiliary objectives. During training z_d is used to predict the domain d and z_y is used to predict the class y for a given input x

$$\mathcal{F}_{\text{DIVA}}(d,x,y) := \mathcal{L}_s(d,x,y) + \mathbb{E}_{q_{\phi_d}(z_d|x)q_{\phi_y}(z_y|x)}\left[\alpha_d \log q_{\omega_d}(d|z_d) + \alpha_y \log q_{\omega_y}(y|z_y)\right], \tag{2}$$

where α_d, α_y are weighting terms for each of these auxiliary objectives. Since our main goal is a domain invariant classifier, during test time we are left with the encoder $q_{\phi_y}(z_y|x)$ and the auxiliary classifier $q_{\omega_y}(y|z_y)$. For predicting the class y of a new input x we only use the mean of z_y.

2.3 Semi-supervised DIVA

In (Kingma et al., 2014) an extension to the VAE framework was introduced that allows us to use labeled as well as unlabeled data during training. While (Kingma et al., 2014) introduced a two step procedure, (Louizos et al., 2015) presented a way of optimizing the decoder of the VAE and the auxiliary classifier jointly. We use the latter approach to learn from supervised data $\{(d_n, x_n, y_n)\}$ as well as from unsupervised data $\{(d_m, x_m)\}$. Analogically to (Louizos et al., 2015), we use $q_{\omega_y}(y|z_y)$ to marginalize out y:

$$\begin{aligned}\mathcal{L}_u(d,x) = \mathbb{E}_{q_{\phi_d}(z_d|x)q_{\phi_x}(z_x|x)q_{\phi_y}(z_y|x)}&[\log p_\theta(x|z_d,z_x,z_y)] \\ &- \beta_d KL(q_{\phi_d}(z_d|x)||p_{\theta_d}(z_d|d)) - \beta_x KL(q_{\phi_x}(z_x|x)||p(z_x)) \\ &+ \beta_y \mathbb{E}_{q_{\phi_y}(z_y|x)q_{\omega_y}(y|z_y)}[\log p_{\theta_y}(z_y|y) - \log q_{\phi_y}(z_y|x)] \\ &+ \mathbb{E}_{q_{\phi_y}(z_y|x)q_{\omega_y}(y|z_y)}[\log p(y) - \log q_{\omega_y}(y|z_y)], \end{aligned} \tag{3}$$

where we use Monte Carlo sampling with the reparametrization trick (Kingma & Welling, 2013) for the continuous latents z_d, z_x, z_y and explicitly marginalize over the discrete variable y. The final objective combines the supervised and unsupervised variational lower bound as well as the two auxiliary objectives. By assuming N labeled and M unlabeled data tuples we arrive at the following objective

$$\mathcal{F}_{\text{SS-DIVA}} = \sum_{n=1}^{N} \mathcal{F}_{\text{DIVA}}(x_n, y_n, d_n) + \sum_{m=1}^{M} \mathcal{L}_u(x_m, d_m) + \alpha_d \mathbb{E}_{q_{\phi_d}(z_d|x_m)}[\log q_{\omega_d}(d_m|z_d)]. \tag{4}$$

3 Related Work

Domain adaptation In contrast to the well known task of domain adaptation, in domain generalization, we do not have access to any data from the test domain during training. In addition, state of the art models like (Tran et al., 2018) are generally focusing on a single training domain and a single test domain. DIVA can handle an arbitrary number of training and test domains.

Transfer learning Nowadays, transferring representations from one task to another is the most common usage of transfer learning (Yosinski et al., 2014). Here, the test domain is not necessarily the same as the training domain. In contrast to domain generalization, the representations learned using training domain data is subsequently adjusted using test domain data. We refer the reader to the extensive review of (Pan & Yang, 2010). We find the same conceptual difference as described above, when comparing our approach to the majority of approaches in the meta learning and zero-shot literature.

Multi-task learning (Zamir et al., 2018) (among many) show that sharing parameters among multiple tasks can lead to better performance and can decrease the amount of labeled examples necessary for each task. Even though we make use of multiple tasks our goal is the opposite. The three encoders of DIVA are forced to learn complementary features. For a general introduction to multi-task learning we refer to (Ruder, 2017).

Fairness The goal of fair classification is to learn good representation that at the same time cannot be used to associate a data sample to a certain group (Zemel et al., 2013). The only difference to domain generalization is the intention behind that goal, e.g. to protect groups of individuals vs. being robust to technical variations. Consequently, DIVA is closely related to the fair VAE (Louizos et al., 2015). The major difference being that we are replacing the hierarchical latent space of the fair VAE with a partitioned latent space.

Latent subspaces A series of very recent papers demonstrate the benefits of having a partitioned latent space in a VAE. (Klys et al., 2018) show that two latent supaces can be better disentangled using binary labels and adversarial training. In (Bouchacourt et al.) exist two latent subspaces as well. Here one is used for individual samples and one is shared among samples from the same group. Last, Hsu & Glass (2018) shows that latent subspaces can be used to capture information from different modalities (e.g., image and audio).

4 EXPERIMENTS

4.1 ROTATED MNIST

The construction of the rotated MNIST dataset follows (Ghifary et al., 2015). We sample 100 images from each of the 10 classes from the original MNIST training dataset. The 1000 images are resized to 16×16 pixels. This set of images is denoted $\mathcal{M}_{0^\circ}$. To create 5 additional domains the images in $\mathcal{M}_{0^\circ}$ are rotated by 15, 30, 45, 60 and 75 degrees. In order to evaluate their domain generalization abilities models are trained on 5 domains and tested on the remaining 6th domain, e.g. train on $\mathcal{M}_{0^\circ}$, $\mathcal{M}_{15^\circ}$, $\mathcal{M}_{30^\circ}$, $\mathcal{M}_{45^\circ}$ and $\mathcal{M}_{60^\circ}$, test on $\mathcal{M}_{75^\circ}$. The evaluation metric is the classification accuracy on the test domain. All experiments are repeated 10 times with 10 different seeds, resulting in 10 different datasets. Detailed information about hyperparameters, architecture and training schedule can be found in the Appendix.

4.2 QUALITATIVE DISENTANGLEMENT

First of all, we visualize the three latent spaces z_d, z_x and z_y, to see if DIVA is able to successfully disentangle them. In addition, we want to see if DIVA utilizes z_x in a meaningful way, since it is not directly connected to any downstream task. For the following visualizations we restrict the size of each latent space z_d, z_x and z_y to 2 dimensions. Therefore, we can plot the latent supspaces directly without applying dimensionality reduction. DIVA is trained on 5000 images from 5 domains ($\mathcal{M}_{0^\circ}$, $\mathcal{M}_{15^\circ}$, $\mathcal{M}_{30^\circ}$, $\mathcal{M}_{45^\circ}$ and $\mathcal{M}_{60^\circ}$).

Figure 2 shows 5000 embeddings z_y encoded by $q_{\phi_y}(z_y|x)$. In Figure 2 (left) the colors indicate the 10 different classes of the MNIST dataset. We observe 10 well separated clusters, each corresponding to 1 of the 10 classes. In stark contrast, Figure 2 (right), where the colors indicate the 5 different training domains, shows no such clustering. It appears that z_y is indeed capturing all necessary information for predicting the class y while containing very little information about the domain d.

In Figure 3 we visualize the 2 dimensional latent space for z_d. Each of the 5000 training images is encoded by $q_{\phi_d}(z_d|x)$. Figure 3 (left) shows very little clustering according to the class label y. However, Figure 3 (right), where each color represents a different domain, shows strong clustering. By comparing the position of the embeddings in the latent space and their original images we conclude that $z_d[0]$ is highly correlated with the rotation angle of an MNIST digit. However, $z_d[1]$ is showing 5 distinct clusters where each cluster corresponds to a different domain. Interestingly, the domains are not ordered as expected. This could be caused by interpolation artifacts that are introduced during the rotation of each image. In any case, z_d is showing the opposite behaviour of z_y. It is capturing mainly information about the domain d.

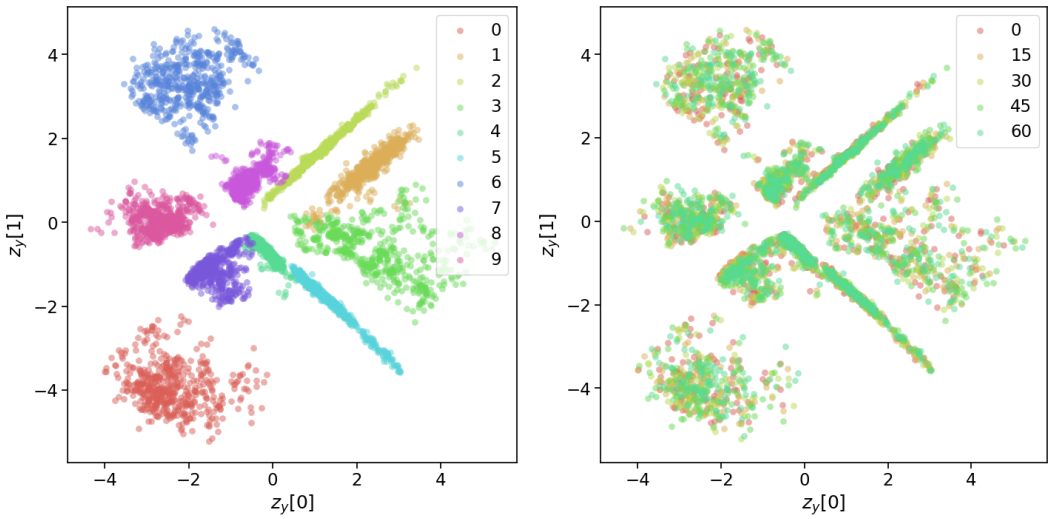

Figure 2: Left: 5000 two-dimensional embeddings z_y encoded by $q_{\phi_y}(z_y|x)$. The color of each point indicates the associated class. Both dimensions of z_y are used to encode the label. Right: Plot of the same embeddings as seen on the left. This time the color indicates the associated domain. No apparent clustering is visible.

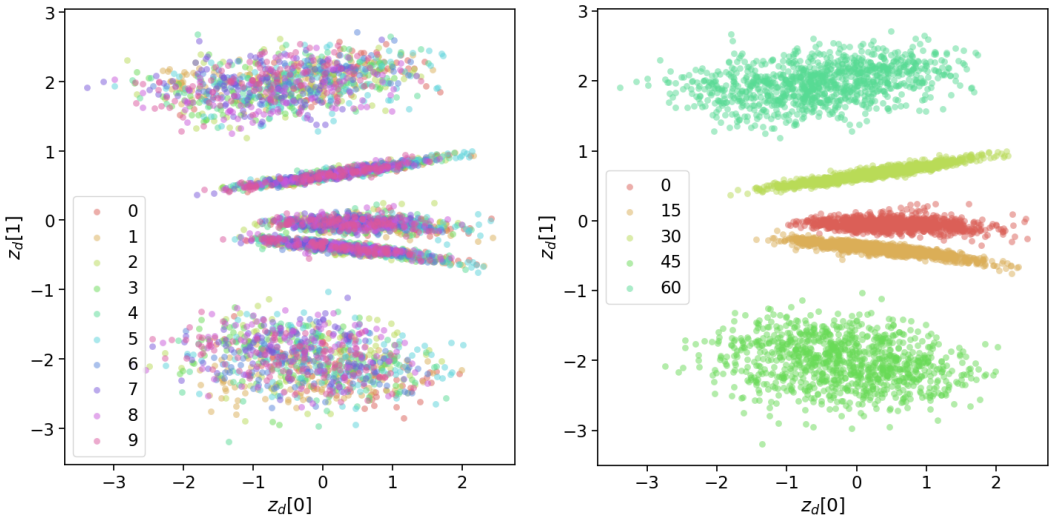

Figure 3: Left: 5000 two-dimensional embeddings z_d encoded by $q_{\phi_d}(z_d|x)$. The color of each point indicates the associated class. No apparent clustering is visible. Right: Plot of the same embeddings as seen on the left. This time the color indicates the associated domain. We observe a high correlation between the rotation angle of each MNIST digit and $z_d[0]$. While in $z_d[0]$ the rotation angle varies smoothly, $z_d[1]$ shows 5 distinct clusters. Each cluster corresponds to a single domain.

In contrast to z_y and z_d, z_x is only used for reconstructing x and has an independent Gaussian prior. In Figure 4 (left) we can see that there is only a small amount of clustering in the lower half of the plot. Again by comparing the position of the embeddings in the latent space and their original images, we try assign meaning to the two axis. Here, $z_x[0]$ is correlated with the line thickness of the MNIST digit. Whereas, $z_x[1]$ corresponds to the width of the whole number, e.g. '1' are usually thin. This explains the high amount of images with class label '1' in the bottom half of Figure 4 (left). In 4 (right) we do not find any clustering.

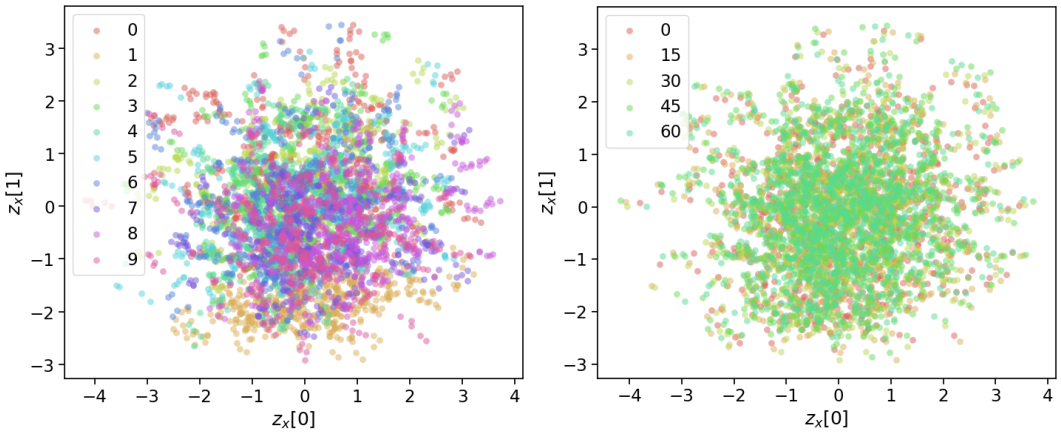

Figure 4: Left: 5000 two-dimensional embeddings z_x encoded by $q_{\phi_x}(z_x|x)$. The color of each point indicates the associated class. We observe a high correlation between the line thickness of each MNIST digit and $z_x[0]$. Whereas, $z_x[1]$ is highly correlated with the width of each MNIST image. For that reason we observe a clustering of embeddings with class '1' at the lower half of the plot. Right: Plot of the same embeddings as seen on the left. This time the color indicates the associated domain. No apparent clustering is visible.

Table 1: We compare DIVA with other state-of-the-art domain generalization methods. Methods in the first half of the table (until the vertical line) use only labeled data. The second half of the table shows results of DIVA when trained semi-supervised (+ X times the amount of unlabeled data). We report the average and standard error of the classification accuracy.

Test	DA	LG	HEX	ADV	DIVA	DIVA(+1)	DIVA(+3)	DIVA(+5)	DIVA(+9)
$\mathcal{M}_{0^\circ}$	86.7	89.7	90.1	89.9	90.2 ± 0.2	91.8 ± 0.2	91.2 ± 0.2	90.6 ± 0.2	90.2 ± 0.1
$\mathcal{M}_{15^\circ}$	98.0	97.8	98.9	98.6	99.0 ± 0.1	98.9 ± 0.1	99.0 ± 0.1	99.0 ± 0.1	99.0 ± 0.1
$\mathcal{M}_{30^\circ}$	97.8	98.0	98.9	98.8	98.5 ± 0.1	98.6 ± 0.1	98.6 ± 0.1	98.7 ± 0.1	98.5 ± 0.1
$\mathcal{M}_{45^\circ}$	97.4	97.1	98.8	98.7	98.3 ± 0.1	98.4 ± 0.1	98.5 ± 0.1	95.5 ± 0.1	98.7 ± 0.1
$\mathcal{M}_{60^\circ}$	96.9	96.6	98.3	98.6	98.3 ± 0.1	98.5 ± 0.1	98.4 ± 0.1	98.5 ± 0.1	98.2 ± 0.1
$\mathcal{M}_{75^\circ}$	89.1	92.1	90.0	90.4	90.8 ± 0.2	91.8 ± 0.2	92.1 ± 0.1	91.9 ± 0.2	91.2 ± 0.2
Avg	94.3	95.3	95.8	95.2	95.9 ± 0.1	96.3 ± 0.1	96.3 ± 0.1	96.2 ± 0.1	96.0 ± 0.1

From these initial qualitative results we conclude that DIVA is disentangling the information contained in x as intended, as z_y is only containing information about y and z_d only information about d. In case of the rotated MNIST dataset z_x captures information about line thickness and digit width, two factors of variation that are not correlated with either the class y or the domain d. We also performed a quantitative analysis of the disentanglement; briefly, we train an MLP classifier that tries to predict y from z_d and z_y from d. The results along with more details can be found in the Appendix.

4.3 Comparison to other methods

We compare our model against the well known domain adversarial neural networks (DA) (Ganin et al., 2015) as well as three recently proposed models: LG (Shankar et al., 2018), HEX (Wang et al., 2019) and ADV (Wang et al., 2019).

For the first half of Table 1 (until the vertical line) we only use labeled data. The first column indicates the rotation angle of the test domain. We report test accuracy on y for all models. For DIVA we report the mean and standard error for 10 repetitions. DIVA achieves the highest accuracy when tested on domain $\mathcal{M}_{0^\circ}$ and $\mathcal{M}_{15^\circ}$. In addition we achieve the highest average test accuracy among all proposed models.

The second half of Table 1 showcases the ability of DIVA to use unlabeled data. For this experiment we add: the same amount (+1) of unlabeled data as well as three (+3), five (+5) and nine (+9) times the amount of unlabeled data to our training set. Here, we first add the unlabeled data to $\mathcal{M}_{0^\circ}$ and

create the data for the other domains as described in Section 4.1. In Table 1 we can clearly see a performance increase when unlabeled data is added to the training set. The effect seems to become smaller when the amount of unlabeled data is much larger than the amount of labeled data as seen in the last column of Table 1.

4.4 ADDITIONAL UNLABELED DOMAINS

In Section 4.3 we show that the performance of DIVA increases when it is presented with additional unlabeled data for each domain. As a result each training domain consists of labeled and unlabeled examples. In this section we investigate a more challenging scenario: we add an additional domain to our training set that consists of only unlabeled examples. Coming back to our introductory example of digital histopathology, here we would we add unlabeled data from a new patient to the training set. In contrast to the experiment in Section 4.3 where we would add unlabeled data from each known patient to the training set.

In the following, we are looking at two different experimental setups: (i) the additional domain is dissimilar to the test domain, e.g. $\mathcal{M}_{30^\circ}$ and $\mathcal{M}_{75^\circ}$, (ii) the additional domain is similar to the test domain, e.g. $\mathcal{M}_{60^\circ}$ and $\mathcal{M}_{75^\circ}$. In both cases we show that DIVA improves when trained with unlabeled data from an unseen domain.

4.4.1 DISSIMILAR DOMAIN

For the first experiment we choose the domains $\mathcal{M}_{0^\circ}$, $\mathcal{M}_{15^\circ}$, $\mathcal{M}_{45^\circ}$ and $\mathcal{M}_{60^\circ}$ to be part of the labeled training set. In addition, unlabeled data from $\mathcal{M}_{30^\circ}$ is used. The test domain is $\mathcal{M}_{75^\circ}$. In Table 2 we can see that even in the case where the additional domain is dissimilar to the test domain DIVA is able to improve.

Table 2: We compare DIVA trained with only labeled data to DIVA trained with additional unlabeled data from $\mathcal{M}_{30^\circ}$. We report the average and standard error of the classification accuracy on $\mathcal{M}_{75^\circ}$.

Test	Only labeled data	Additional unlabeled
$\mathcal{M}_{75^\circ}$	90.0 ± 0.1	91.0 ± 0.1

4.4.2 SIMILAR DOMAIN

For the second experiment we choose the domains $\mathcal{M}_{0^\circ}$, $\mathcal{M}_{15^\circ}$, $\mathcal{M}_{30^\circ}$ and $\mathcal{M}_{45^\circ}$ to be part of the labeled training set. In addition, unlabeled data from $\mathcal{M}_{60^\circ}$ is used. The test domain is $\mathcal{M}_{75^\circ}$. When comparing the results in Table 3 to the results in Table 1 and 2 we notice a drop in accuracy of about 20% for DIVA trained with only labeled data. However, when trained with unlabeled data from $\mathcal{M}_{60^\circ}$ the drop in accuracy is less than 2%.

Table 3: We compare DIVA trained with only labeled data to DIVA trained with additional unlabeled data from $\mathcal{M}_{60^\circ}$. We report the average and standard error of the classification accuracy on $\mathcal{M}_{75^\circ}$.

Test	Only labeled data	Additional unlabeled
$\mathcal{M}_{75^\circ}$	69.2 ± 0.3	89.4 ± 0.1

5 CONCLUSION

We have proposed the DIVA model as a generative model with three independent sources of variation. We demonstrated through quantitative and qualitative experiments on rotated MNIST that our model successfully learns representations z_y that are invariant with respect to the domain d. In future work, we want to evaluate DIVA on a more complex dataset.

ACKNOWLEDGMENTS

REFERENCES

Aharon Azulay and Yair Weiss. Why do deep convolutional networks generalize so poorly to small image transformations? *arXiv:1805.12177 [cs]*, May 2018. arXiv: 1805.12177.

Rianne van den Berg, Leonard Hasenclever, Jakub M. Tomczak, and Max Welling. Sylvester Normalizing Flows for Variational Inference. *arXiv:1803.05649 [cs, stat]*, March 2018. arXiv: 1803.05649.

Gilles Blanchard, Gyemin Lee, and Clayton Scott. Generalizing from Several Related Classification Tasks to a New Unlabeled Sample. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 24*, pp. 2178–2186. Curran Associates, Inc., 2011.

Diane Bouchacourt, Ryota Tomioka, and Sebastian Nowozin. Multi-Level Variational Autoencoder: Learning Disentangled Representations from Grouped Observations. pp. 8.

Ricky T. Q. Chen, Xuechen Li, Roger Grosse, and David Duvenaud. Isolating Sources of Disentanglement in Variational Autoencoders. *arXiv:1802.04942 [cs, stat]*, February 2018. arXiv: 1802.04942.

Francesco Ciompi, Oscar Geessink, Babak Ehteshami Bejnordi, Gabriel Silva de Souza, Alexi Baidoshvili, Geert Litjens, Bram van Ginneken, Iris Nagtegaal, and Jeroen van der Laak. The importance of stain normalization in colorectal tissue classification with convolutional networks. *arXiv:1702.05931 [cs]*, February 2017. arXiv: 1702.05931.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, Franois Laviolette, Mario Marchand, and Victor Lempitsky. Domain-Adversarial Training of Neural Networks. *arXiv:1505.07818 [cs, stat]*, May 2015. arXiv: 1505.07818.

Muhammad Ghifary, W. Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. Domain Generalization for Object Recognition with Multi-task Autoencoders. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 2551–2559, Santiago, Chile, December 2015. IEEE. ISBN 978-1-4673-8391-2. doi: 10.1109/ICCV.2015.293.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. pp. 22, 2017.

Irina Higgins, David Amos, David Pfau, Sebastien Racaniere, Loic Matthey, Danilo Rezende, and Alexander Lerchner. Towards a Definition of Disentangled Representations. *arXiv:1812.02230 [cs, stat]*, December 2018. arXiv: 1812.02230.

Wei-Ning Hsu and James Glass. Disentangling by Partitioning: A Representation Learning Framework for Multimodal Sensory Data. *arXiv:1805.11264 [cs, eess, stat]*, May 2018. arXiv: 1805.11264.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. *arXiv:1312.6114 [cs, stat]*, December 2013. arXiv: 1312.6114.

Diederik P. Kingma, Danilo J. Rezende, Shakir Mohamed, and Max Welling. Semi-Supervised Learning with Deep Generative Models. *arXiv:1406.5298 [cs, stat]*, June 2014. arXiv: 1406.5298.

Jack Klys, Jake Snell, and Richard Zemel. Learning Latent Subspaces in Variational Autoencoders. *arXiv:1812.06190 [cs, stat]*, December 2018. arXiv: 1812.06190.

Maxime W. Lafarge, Josien P. W. Pluim, Koen A. J. Eppenhof, Pim Moeskops, and Mitko Veta. Domain-adversarial neural networks to address the appearance variability of histopathology images. *arXiv:1707.06183 [cs]*, 10553, 2017. doi: 10.1007/978-3-319-67558-9. arXiv: 1707.06183.

Christos Louizos, Kevin Swersky, Yujia Li, Max Welling, and Richard Zemel. The Variational Fair Autoencoder. *arXiv:1511.00830 [cs, stat]*, November 2015. arXiv: 1511.00830.

Lars Maaloe, Marco Fraccaro, Valentin Livin, and Ole Winther. BIVA: A Very Deep Hierarchy of Latent Variables for Generative Modeling. *arXiv:1902.02102 [cs, stat]*, February 2019. arXiv: 1902.02102.

Krikamol Muandet, David Balduzzi, and Bernhard Schlkopf. Domain Generalization via Invariant Feature Representation. *arXiv:1301.2115 [cs, stat]*, January 2013. arXiv: 1301.2115.

Sinno Jialin Pan and Qiang Yang. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, October 2010. ISSN 1041-4347. doi: 10.1109/TKDE.2009.191.

Sebastian Ruder. An Overview of Multi-Task Learning in Deep Neural Networks. *arXiv:1706.05098 [cs, stat]*, June 2017. arXiv: 1706.05098.

Shiv Shankar, Vihari Piratla, Soumen Chakrabarti, Siddhartha Chaudhuri, Preethi Jyothi, and Sunita Sarawagi. Generalizing Across Domains via Cross-Gradient Training. *arXiv:1804.10745 [cs, stat]*, April 2018. arXiv: 1804.10745.

Antonio Torralba and Alexei A. Efros. Unbiased look at dataset bias. In *in CVPR*, 2011.

Luan Tran, Kihyuk Sohn, Xiang Yu, Xiaoming Liu, and Manmohan Chandraker. Joint Pixel and Feature-level Domain Adaptation in the Wild. *arXiv:1803.00068 [cs]*, February 2018. arXiv: 1803.00068.

Haohan Wang, Zexue He, Zachary C Lipton, and Eric P Xing. LEARNING ROBUST REPRESENTATIONS BY PROJECT- ING SUPERFICIAL STATISTICS OUT. pp. 16, 2019.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *arXiv:1411.1792 [cs]*, November 2014. arXiv: 1411.1792.

Amir Zamir, Alexander Sax, William Shen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling Task Transfer Learning. *arXiv:1804.08328 [cs]*, April 2018. arXiv: 1804.08328.

Richard Zemel, Yu Wu, Kevin Swersky, Toniann Pitassi, and Cynthia Dwork. Learning fair representations. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, pp. III–325–III–333. JMLR.org, 2013.

APPENDIX

TRAINING PROCEDURE

All DIVA models are trained for 500 epochs. The training is terminated if the classification loss for y has not improved for 100 epochs. As proposed in (Higgins et al., 2017), we linearly increase β_d, β_x and β_y from 0.0 to 1.0 during the first 100 epochs of training. We set α_d = 50. As seen in (Maaloe et al., 2019), we adjust α_y according to the ratio of labeled (N) and unlabeled data (M),

$$\alpha_y = \gamma \frac{M+N}{N}, \tag{5}$$

where we set $\gamma = 100$. Last, z_d, z_x and z_y each have 32 latent dimensions. All hyperparameters were determined by training DIVA on $\mathcal{M}_{0^\circ}$, $\mathcal{M}_{15^\circ}$, $\mathcal{M}_{30^\circ}$, $\mathcal{M}_{45^\circ}$ and testing on $\mathcal{M}_{60^\circ}$.

ARCHITECTURE

The encoder and decoder architecture is taken from (Berg et al., 2018).

Table 4: Architecture for $p_\theta(x|z_d, z_x, z_y)$. The parameters for ConvTranspose2d are: output channels, kernel size, stride, padding, output padding. The parameters for Conv2d are: output channels, kernel size, stride, padding.

block	details
1	ConvTranspose2d(64, 4, 1, 0, 0), BatchNorm2d(64), LeakyReLU()
2	ConvTranspose2d(64, 5, 1, 2, 0), BatchNorm2d(64), LeakyReLU()
3	ConvTranspose2d(32, 5, 2, 2, 1), BatchNorm2d(32), LeakyReLU()
4	ConvTranspose2d(32, 5, 1, 2, 0), BatchNorm2d(32), LeakyReLU()
5	ConvTranspose2d(32, 5, 2, 2, 1), BatchNorm2d(32), LeakyReLU()
6	ConvTranspose2d(32, 5, 1, 2, 0), BatchNorm2d(32), LeakyReLU()
7	Conv2d(256, 5, 1, 2)
7	Conv2d(256, 1, 1, 0)

Table 5: Architecture for $p_{\theta_d}(z_d|d)$ and $p_{\theta y}(z_y|y)$. Each network has two heads one for the mean and one for the scale. The parameter for Linear is: output features.

block	details
1	Linear(32), BatchNorm1d(32), LeakyReLU(), Dropout(0.2)
2.1	Linear(32)
2.2	Linear(32), Softplus()

Table 6: Architecture for $q_{\phi_d}(z_d|x)$, $q_{\phi_x}(z_x|x)$ and $q_{\phi_y}(z_y|x)$. Each network has two heads one for the mean one and for the scale. The parameters for Conv2d are: output channels, kernel size, stride, padding. The parameter for Linear is: output features.

block	details
1	Conv2d(32, 5, 1, 2), BatchNorm2d(32), LeakyReLU()
2	Conv2d(32, 5, 2, 2), BatchNorm2d(32), LeakyReLU()
3	Conv2d(64, 5, 1, 2), BatchNorm2d(64), LeakyReLU()
4	Conv2d(64, 5, 2, 2), BatchNorm2d(64), LeakyReLU()
5	Conv2d(64, 5, 1, 2), BatchNorm2d(64), LeakyReLU()
6	Conv2d(256, 4, 1, 0), BatchNorm2d(256), LeakyReLU()
7.1	Linear(32)
7.2	Linear(32), Softplus()

Table 7: Architecture for $q_{\omega_d}(d|z_d)$ and $q_{\omega_y}(y|z_y)$. The parameter for Linear is: output features.

block	details
1	Linear(32), BatchNorm1d(32), LeakyReLU(), Dropout(0.2)
2	Linear(5 for $q_{\omega_d}(d\|z_d)$/10 for $q_{\omega_y}(y\|z_y)$)

QUANTITATIVE DISENTANGLEMENT

While Section 4.2 contains only qualitative results, we now quantify the disentanglement and its influence on the classification accuracy. A good indicator for disentanglement is to test how predictive z_d is of y and z_y is of d. First, we train DIVA using the training data from $\mathcal{M}_{0^\circ}$, $\mathcal{M}_{15^\circ}$, $\mathcal{M}_{30^\circ}$, $\mathcal{M}_{45^\circ}$ and $\mathcal{M}_{60^\circ}$. After training DIVA, we use $q_{\phi_d}(z_d|x)$ and $q_{\phi_y}(z_y|x)$ to generate embeddings z_d and z_y for every x. Finally we train a two layer MLP for 500 epochs for the following 4 cases: predicting y from z_d, predicting d from z_d, predicting y from z_y and predicting d from z_y.

We repeat this experiment 3 times for $\beta_d = \beta_x = \beta_y = 1$, $\beta_d = \beta_x = \beta_y = 5$ and $\beta_d = \beta_x = \beta_y = 10$. The first thing we notice in Table 8 is that the two layer MLP is able to overfit perfectly when predicting d from z_d and y from z_y for all choices of β_d, β_x and β_y. When we try to predict the class label y from z_d the accuracy is capped at around 26%, while the random chance accuracy for this task is 10%. In the other case, where we try to predict the domain label d from z_y the accuracy is capped at around 41%. The random chance accuracy for this task is 20%. In both cases we observe that the classification accuracy decreases if we increase β_d, β_x and β_y. This trend implies that for higher βs z_d becomes less predictive of y and z_y becomes less predictive of d, i.e. z_y and z_d are better disentangled. This finding aligns with observations made in (Higgins et al., 2017).

Table 8: Classification results for embeddings of the training set for different values of β_d, β_x and β_y.

Task	$\beta = 1$	$\beta = 5$	$\beta = 10$
y from z_d	26.6 $\pm$ 0.1	26.5 $\pm$ 0.1	26.0 $\pm$ ¿0.1
d from z_d	100.0 $\pm$ 0.0	100.0 $\pm$ 0.0	100.0 $\pm$ 0.0
y from z_y	100.0 $\pm$ 0.0	100.0 $\pm$ 0.0	100.0 $\pm$ 0.0
d from z_y	41.4 $\pm$ 0.1	41.0 $\pm$ 0.1	40.8 $\pm$ ¿0.1

In addition, we look at the classification performance on the test domain $\mathcal{M}_{75^\circ}$. In Table 9 we notice that test accuracy decreases when we increase β_d, β_x and β_y. By comparing this observation with the results in Table 8, we conclude that a better disentangled latent space results in worse test accuracy on the rotated MNIST dataset. This finding aligns well with observation made in (Wang et al., 2019), that some information about the domain seems to be useful for predicting the class in case of the rotated MNIST dataset.

Table 9: Classification results on the test domain $\mathcal{M}_{75^\circ}$ for different values of β_d, β_x and β_y.

Test	$\beta = 1$	$\beta = 5$	$\beta = 10$
$\mathcal{M}_{75^\circ}$	90.9 $\pm$ 0.2	89.4 $\pm$ 0.2	88.9 $\pm$ 0.2

CONDITIONAL GENERATION

Yet another way to gain insight into the disentanglement abilities of DIVA is conditional generation. We first train DIVA with $\beta_d = \beta_x = \beta_y = 10$ using $\mathcal{M}_{0^\circ}$, $\mathcal{M}_{15^\circ}$, $\mathcal{M}_{30^\circ}$, $\mathcal{M}_{45^\circ}$ and $\mathcal{M}_{60^\circ}$ as training domains. After training we perform two experiments. In the first one we are fixing the class and varying the domain. In the second experiment we are fixing the domain and varying the class. Notice, the overall quality of images in this dataset is rather poor since they are rescaled to 16×16 pixels.

Change of class The first row of Figure 5 (left) shows the input images x for DIVA. First, we generate embeddings z_d, z_x and z_y for each x using $q_{\phi_d}(z_d|x)$, $q_{\phi_x}(z_x|x)$ and $q_{\phi_y}(z_y|x)$. Second, we replace z_y with a sample z'_y from the conditional prior $p_{\theta_y}(z_y|y)$. Last, we generate new images from z_d, z_x and z'_y using the trained encoder $p_\theta(x|z_d, z_x, z_y)$. In Figure 5 (left) rows 2 to 11 correspond to the classes '0' to '9'. We observe that the rotation angle (encoded in z_d) and the line thickness (encoded in z_x) are well preserved, while the class of the image is changing as intended.

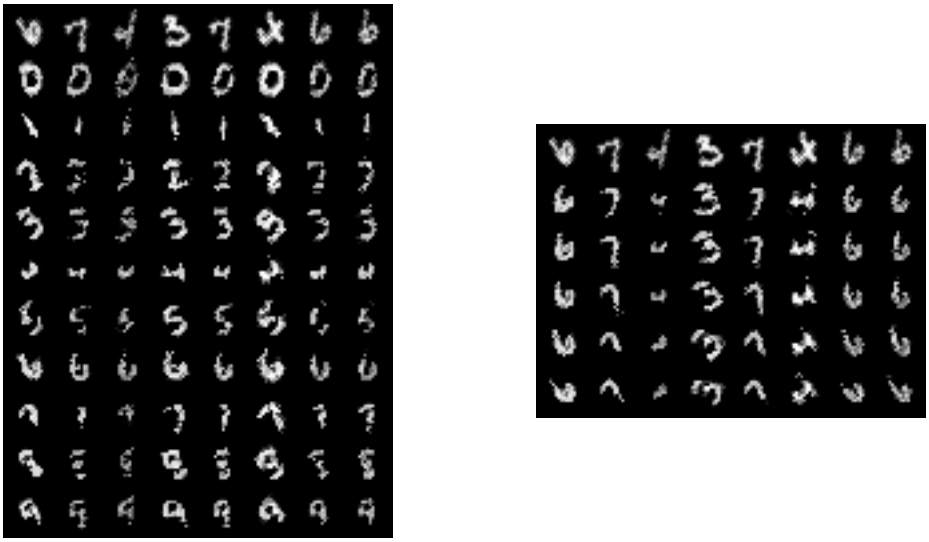

Figure 5: Reconstructions. First row is input, row 2 to 11 correspond to labels 0 to 9.. Reconstructions. First row is input, row 2 to 6 correspond to domains 0, 15, 30, 45, 60.

Change of domain We repeat the experiment from above but this time we keep z_x and z_y fixed while changing the domain. After generating embeddings z_d, z_x and z_y for each x in the first row of Figure 5 (right), we replace z_d with a sample z'_d from the conditional prior $p_{\theta_d}(z_d|d)$. Finally, we generate new images from z'_d, z_x and z_y using the trained encoder $p_\theta(x|z_d, z_x, z_y)$. In Figure 5 (right) rows 2 to 6 correspond to the domains $\mathcal{M}_{0^\circ}$ to $\mathcal{M}_{60^\circ}$. Again, DIVA shows the desired behaviour: While the rotation angle is changing the class and style of the original image is maintained.