INVARIANT BATCH NORMALIZATION FOR MULTI-SOURCE DOMAIN GENERALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

We consider the domain generalization problem, where the test domain differs from the training domain. For deep neural networks, we show that the batch normalization layer is a highly unstable component under such domain shifts, and we identify two sources for its instability. Based on this observation, we propose a new learning formulation that can learn robust neural networks so that the corresponding batch normalization layers are invariant under domain shifts. Experimental results on three standard domain generalization benchmarks demonstrate that our method can learn neural network models with significantly more stable batch normalization layers on unseen domains, and the improved stability leads to superior generalization performances.

1 INTRODUCTION

Conventional neural networks are based on the assumption that the training and test data are from the same underlying distribution. However, this assumption does not always hold in real scenarios, which causes the test performance to drop dramatically in case of distribution shifts. In this work, we focus on the domain generalization task which aims to relax the i.i.d assumption between the training and test data, without accessing test domain data. Unlike the traditional domain adaptation setting, domain generalization tasks assume that we can not access the test domain during training, which requires that the learned model should have a good generalization ability to fit well on the unseen distribution shifts. It is widely believed that the model that is robust to the probable domain shifts should rely on the domain invariant features, rather than the domain style features. To extract domain invariant features, most of the existing algorithms utilize the multi-source domain information to guide the training. Our algorithm also follows this setting.

Domain styles, including object textures, lighting conditions and filter effects, are considered as a major source of distribution shifts. Several approaches (Arjovsky et al., 2019; Dou et al., 2019; Li et al., 2018a; Balaji et al., 2018), including distribution matching based, meta-learning based are proposed to learn an invariant representation with discarding the spurious correlation domain factor. However, most of the work focuses on using the final classifier to measure the invariant of the learned representation, which may not effectively identify the domain style information. Recent work (Li et al., 2018e; Wang et al., 2019; Pan et al., 2018; Nam & Kim, 2018) illustrates that batch normalization is extremely unstable under domain shifts. Studies (Huang & Belongie, 2017; Karras et al., 2019) on style transfer show that normalization layers are highly correlated with image style, motivating us to analyze the instability of batch normalization in the distribution shift.

In particular, we conduct an empirical study and observe that the reasons of the instability in batch normalization are due to two distribution shifts: 1). marginal distribution shift in the learned representation makes domains have different normalization statistics, 2). conditional distribution shift leads to the instability of the rescaling parameters in normalization. In the unsupervised domain adaptation field, the marginal distribution shift is addressed by replacing normalization statistics from the source domain to the target domain (Li et al., 2018e; Wang et al., 2019). However, in domain generalization, target domain data is unseen during training, making replace normalization statistics impossible. To alleviate the distribution shift in the multi-source domain generalization scenario, we propose a learning formulation that stabilizes the batch normalization layer by learning an invariant representation. Specifically, our formulation identifies the domain style information by measuring the performance gap between the shared and domain optimal normalization layer. If

some spurious features of the representation layer are related to a domain, we can always train a better domain-specific model by merely adapting the normalization layer bias to a specific domain. And the features with different conditional distribution can be removed by forcing that the shared normalization layer is also optimal in each specific domain. The features with marginal distribution shifts are discarded by penalizing the performance drop when replacing the domain optimal normalization statistics to domain mixture. Through ensuring that the domain shared normalization is also optimal in each domain, the representation layer is explicitly guided to learn robustness features, leading to an insensitive neural network under distribution shift.

We summarize our contributions as follows:

- From empirical analysis, we show batch normalization is highly unstable under domain shifts, and we identify two sources of domain sensitivities for batch normalization: normalization parameter sensitivity due to the change of distribution statistics, and rescaling parameter sensitivity due to the change of conditional distribution.
- We present a new learning formulation that can be used to learn batch normalization parameters that are insensitive to domain changes. This formulation learns neural network models with robust feature representations, so that batch normalization layer becomes invariant under unknown distribution shifts. This property improves the model's generalization performance on unseen domains.
- We conduct extensive experiments on standard domain generalization benchmarks to show that our method successfully learns robust batch normalization that achieves state-of-the-art performances on these datasets.

2 Related work

In this section, we briefly review the topics relevant to our work.

2.1 Domain generalization

Domain generalization aims to enhance the model generalization ability under the distribution shift. Unlike conventional domain adaptation setting, domain generalization can not access target domain data during training, which requires the model to learn an invariant representation under the distribution shift.

Distribution matching. Matching distribution in latent representation is a widely surveyed topic in the domain adaptation field (Ben-David et al., 2010; Ganin et al., 2016), which is utilized to alleviate the domain shift. Specifically, conventional domain adaptation methods learned a domain invariant representation by matching the latent marginal distribution with minimizing the MMD distance or adversarial training. (Li et al., 2018c;d) further erase the domain style information by aligning both marginal and conditional distribution in multiple source domain scenarios. (Li et al., 2018b) improves MMD based method with Adversarial Autoencoder framework. Later, IRM (Arjovsky et al., 2019) provides a new paradigm in evaluating invariant: *"An invariant representation $\Phi(X)$ is one such that the optimal linear predictor, ω is the same across all environments"*. (Arjovsky et al., 2019) converts the invariant condition to a bi-level optimization problem and solves it by penalizing the gradient norm in domain direction. Although the gradient norm represents the distance to the optimal point, it is scale sensitive to the parameter, making the penalty weight hard to tune. Our approach follows IRM in measuring the invariant of the learned representation, while provides a flexible formulation to stabilize normalization. There is work improving IRM, like (Ahuja et al., 2020) which further relaxes the linear classifier and transformation assumption in IRM with playing an ensemble game among environments. Our approach is orthogonal to them.

Other approaches. Except for matching the distribution, meta-learning, data augmentation, decomposition and other related methods are also proposed to increase the model generalization ability. Meta-learning aims at improving the model's transferability by giving the experience of multiple learning episodes, which has a similar training pipeline with domain generalization. To learn a domain invariant model, (Li et al., 2018a; 2019) adapt MAML (Finn et al., 2017) by splitting the multiple training domains with meta-train and meta-test split. (Balaji et al., 2018; Dou et al., 2019) further improve MLDG (Li et al., 2018a) by generating a meta-based regularization with minimizing

domain based discrepancy that maximizes the meta-test domain likelihood. (Li et al., 2017; Piratla et al., 2020) propose to decompose the learned parameters with low-rank structure and remove the domain bias information. Augmenting data in the domain direction is another effective approach to regularize the model, which including generating domain adversarial examples (Shankar et al., 2018; Volpi et al., 2018) and transferring the image style in different domains (Gong et al., 2019; Nam et al., 2019). (Tseng et al., 2020; Lee et al., 2020) perturb latent representation to enhance the robustness across tasks and distributions.

2.2 NORMALIZATION

Normalization layer is a common technique used to stabilize and accelerate the training in neural network. Batch normalization (Ioffe & Szegedy, 2015) is the first pioneering work, which normalizes the latent layer along the batch and feature dimension. Existed approaches (Ioffe & Szegedy, 2015; Li et al., 2018e; Wang et al., 2019) claim that batch normalization suffers from the distribution shift due to the change of distribution statistics. (Li et al., 2018e) addresses the statistics shift by replacing source domain statistics with the target domain. (Pan et al., 2018; Nam & Kim, 2018) alleviate the statistics shift by inserting a instance normalization before batch normalization to remove image style, however, it may sacrifice the model's discriminative. In this work, we further analyze the influence of the distribution shift on batch normalization, where marginal distribution shift leads to the change of normalization statistics and conditional distribution shift leads to the sensitivity of the rescaling parameter. The success of style transfer by training the rescaling parameter also illustrates the effect of conditional distribution shift on the normalization layer (Huang & Belongie, 2017; Karras et al., 2019). (Seo et al., 2019; Chang et al., 2019; Xie et al., 2020) address the sensitivity of normalization layer by augmenting its parameter space with domain-specific, which largely improves the training performance. However, augmenting parameter space can not learn a stable normalization layer, which makes them can not apply to the unseen domain. Our proposed InvarNorm elicit a stable batch normalization layer by learning robust representation with less to marginal and conditional distribution shift across domains.

3 APPROACH

In this section, we first analyze the source of unstable in batch normalization under distribution shift. Then we introduce our Invariant Batch Normalization method to learn an invariant representation with domain stable normalization layer.

3.1 BATCH NORMALIZATION

Let $\{(x_i, y_i)\}_{i=1}^m$ denotes the data in the batch, and μ_b and σ_b are the mean and variance. During training stage, batch normalization calculates the normalization statistics for each feature channel along the batch dimension and uses the calculated statistics to normalize the feature to $\mathcal{N}(0, 1)$. After normalizing, learned rescaling parameter γ and β are used to rescale the features, which is stated as follows:

$$\mu_b = \frac{1}{m}\sum_{i=1}^{m} x_i, \quad \sigma_b = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_b)^2, \quad \hat{x} = \frac{x - \mu_b}{\sqrt{\sigma_b^2 + \epsilon}}, \quad y = \gamma\hat{x} + \beta, \tag{1}$$

During the inference stage, BN uses the moving average μ_b and σ_b from the training data for predicting the test data. It is proper if the train and test data are drawn from $i.i.d.$. However, in the multi-domain setting, the $i.i.d.$ assumption is violated due to the distribution shift across different domains. Let (X^d, Y^d) denotes the data from domain d. We use superscript s and t to denote source and target domain, respectively. The distribution shift means that $p(X^s, Y^s) \neq p(X^t, Y^t)$.

3.2 SOURCE OF DOMAIN UNSTABLE IN BATCH NORMALIZATION

Following traditional studies in transfer learning, we analyze the source of performance drop is from two distribution shifts: marginal distribution and conditional distribution shift in the learned feature representation. Specifically, we argue that the distribution shifts affect the normalization layer from two perspectives:

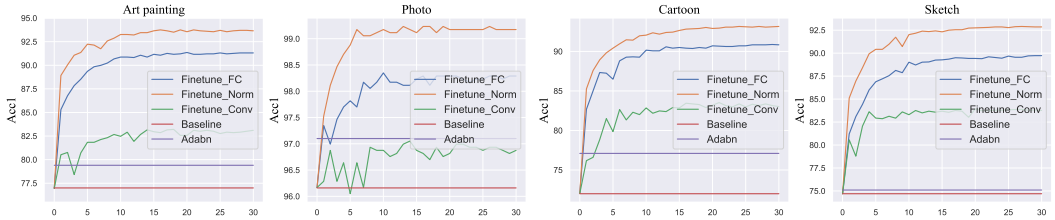

Figure 1: Empirical study with Adabn and fine-tuning different parameters with the same parameter size on PACS dataset, where FC, Norm, conv1 denotes the final classifier , normalizaiton layer and the first convolution layer respectively.

- The different of the marginal distribution between source and target makes the normalization statistics in hidden representation layer change, leading to inconsistent between training and inference: $P(\Phi(X^s)) \neq P(\Phi(X^t)) \rightarrow \mu_b^s \neq \mu_b^t, \sigma_b^s \neq \sigma_b^t$, where Φ denotes the feature encoder before the normalization layer. To verify if this distribution shift causes the performance drop, we check the performance gain after adjusting the normalization statistics by AdaBN (Li et al., 2018e).
- If the conditional distributions of image and label is different between source and target, γ and β in rescaling function would be different, which leads to the performance drop when applying the source learned γ and β to the target domain. $P(Y^s|\Phi(X^s)) \neq P(Y^t|\Phi(X^t)) \rightarrow \gamma^s \neq \gamma^t, \beta^s \neq \beta^t$. We verify this distribution shift by measuring the performance improvement after fine-tuning the γ and β in the target domain data.

In Figure 1, we display the empirical study about BN on the PACS dataset (Li et al., 2017). By comparing the source only model with AdaBN, we can find that aligning the normalization statistics consistently improve the trained model 1-2 % accuracy rate in the target domain, illustrating the existence of statistics shift and performance drop. By comparing AdaBN with the model fine-tuned the parameters γ and β while fixing the other layer, we get that the conditional distribution shift in normalization has a tremendous influence on performance degradation. What's more, we also compared the performance gain of fine-tuning different parameters (First convolution layer, BN layer, Final classifier layer) to find the most domain sensitive parameters. We can observe that fine-tuning the batch normalization parameter has the fastest convergence speed and the largest performance gain, proving that BN is more sensitive compared with other parameters.

3.3 Invariant Normalization

After identifying the source of the unstable in the normalization layer, we introduce the way we measure its stability and enhance it. Following the recent work (Arjovsky et al., 2019; Ahuja et al., 2020) that, if the learned representation layer is invariant, the shared classifier is optimal in each specific domain, which is formulated as follows:

$$R^d(\mathcal{L}(f_{\Phi,\omega}(x^d), y^d)) \leq \min_{\omega_d} R^d(\mathcal{L}(f_{\Phi,\omega_d}(x^d), y^d)) + \epsilon_d \quad \forall d \in \mathcal{E}_{\text{tr}}, \tag{2}$$

where R is the empirical risk, $\mathcal{L}$ is the loss function, Φ is the feature encoder that maps the input data into latent representation $\mathcal{X} \rightarrow \mathcal{Z}$, ω, ω_d are the shared and domain-specific classifiers in the training domains $\mathcal{E}_{\text{tr}}$ and ϵ_d is a small positive number. In our Invariant Batch Normalization, we treat the convolutional layer with parameter θ as the feature encoder Φ and the normalization layer with parameter γ and β as the classifier ω. If our normalization layer satisfies the inequality in Formula 2, the stability of normalization are ensured since one shared normalization can achieve optimal in each specific domain.

To sum up, we require our neural network not only achieves small empirical risk but also has a stable normalization layer, which can be phrased as follows:

$$\begin{aligned} &\min_{\Phi,\omega} \sum_{d \in \mathcal{E}_{\text{tr}}} R^d(\mathcal{L}(f_{\Phi,\omega}(x^d), y^d)) \\ &s.t. \quad R^d(\mathcal{L}(f_{\Phi,\omega}(x^d), y^d)) \leq \min_{\omega_d} R^d(\mathcal{L}(f_{\Phi,\omega_d}(x^d), y^d)) + \epsilon_d \quad \forall d \in \mathcal{E}_{\text{tr}}. \end{aligned} \tag{3}$$

To solve this objective function, we convert the objective function in Formula 3 with a new practical formulation.

$$\min_{\Phi,\omega}\max_{\omega_d}\sum_{d\in\mathcal{E}_{\text{tr}}} R^d(\mathcal{L}(f_{\Phi,\omega}(x^d),y^d)) + \lambda\left(R^d(\mathcal{L}(f_{\Phi,\omega}(x^d),y^d)) - R^d(\mathcal{L}(f_{\Phi,\omega_d}(x^d),y^d))\right), \quad (4)$$

where λ is the penalty weight that balances the discriminative ability (ERM term) with the stable of the normalization layer (penalty term). In Formula 4, we maximize $R^d(\mathcal{L}(f_{\Phi,\omega_d}(x^d),y^d)$ with respect to the parameter in the domain-specific normalization layer ω_d to extract the domain bias information, and minimize $\left(R^d(\mathcal{L}(f_{\Phi,\omega}(x^d),y^d)) - R^d(\mathcal{L}(f_{\Phi,\omega_d}(x^d),y^d))\right)$ with respect to the parameter Φ in the shared convolution layer to remove the domain bias information extracted by ω_d. By minimizing the performance gap between the shared normalization layer and the domain optimal normalization layer, the stability of the shared normalization on the given representation can be guaranteed due to achieving optimal in each specific domain.

Marginal distribution shift

Except for having a stable rescaling parameter γ and β, an invariant batch normalization should make the normalization statistics μ and σ insensitive under marginal distribution shift. In multi-domain setting, we achieve this goal by penalizing the different of domain-specific normalization statistics with the average of them. Specifically, we use the domain-specific normalization statistics in calculating the ERM term as in (Xie et al., 2020; Seo et al., 2019). And in the penalty term, we compute the shared part by using the average of the statistics, which can lead to performance drop due to the different of the normalization statistics. Our algorithm alleviates this marginal distribution shifts by penalizing the performance drop compared with using domain-specific and domain shared normalization statistics.

By penalizing the change of normalization statistics and rescaling parameter between shared and domain-specific branch, the learned representation is guided to only extract those invariant features that have less shift in marginal and conditional distribution. With learning an invariant representation, batch normalization is insensitive under distribution shifts.

3.4 TRAINING PIPELINE

We summarize the training pipeline in Algorithm 3.4. During inference, the domain-specific components are discarded and the shared component are used to forward the input data.

Algorithm 1 Training pipeline for Invariant Normalization

Input: Source domain training data $\{(x_i^d, y_i^d)\}_{i=1}$ with $d \in \mathcal{E}_{\text{tr}}$, Shared components Φ, ω, and domain-specific components ω_d, penalty weight λ.
while Training **do**
  Sample data from multiple domains and compute the first ERM term.
  Update shared components Φ and ω.
  Fix feature encoder Φ
  while Fine-tune domain-specific component **do**
    Sample data from multiple domains and compute $R^d(\mathcal{L}(f_{\Phi,\omega_d}(x^d),y^d)$
    Update domain-specific component ω_d
  end while
  Compute penalty term $\lambda\left(R^d(\mathcal{L}(f_{\Phi,\omega}(x^d),y^d)) - R^d(\mathcal{L}(f_{\Phi,\omega_d}(x^d),y^d))\right)$
  Update feature encoder Φ
end while

3.5 DISCUSSION WITH IRM

Our approach is motivated by IRM and share some similarity in measuring the invariant of the learned neural network. However, our formulation differs to IRM in solving the objective function. Compared with IRM that use the gradient norm, we propose a scale-invariant minmax formulation, which makes the penalty weight less sensitive to the change of neural network. Our formula also can be applied in other environments. We left it as our future work.

4 EXPERIMENTS

In this section, we conduct extensive experiments on three domain generalization datasets PACS (Li et al., 2017), Office-home (Venkateswara et al., 2017), VLCS (Fang et al., 2013) to evaluate the generalization ability of our proposed Invariant Batch Normalization.

4.1 EXPERIMENTAL SETUP

We train our model on source domain datasets and leave out one target domain as the unseen domain to evaluate its domain generalization ability. To avoid the leakage of unseen domain information during training, we select the final model from the training domain validation set. In this work, we implement our algorithm in the ResNet-18 and ResNet-50 network He et al. (2016) and initialized it with ImageNet pre-trained weight. SGD optimizer with learning rate 0.005 and weight decay 1e-4 is utilized to optimize the network. The batch size is set as 32 for each training domain with 4,000 training steps in PACS and VLCS dataset and 6,000 training steps in Office-home dataset.

We compare our InvarNorm with the following state-of-the-art methods: **ERM** is our baseline method that aggregates all the source domains data and optimizes with the empirical risk minimization. **IRM** (Arjovsky et al., 2019) is a regularization based method that learns an invariant representation by removing the domain style information in the latent representation. Our method shares some similarities with it in removing the domain style information, but our method focus on addressing the invariant problem in normalization layer and utilize a better objective function that is fixed with our framework. **JiGen** (Carlucci et al., 2019) is a data augmentation based approach which augments the data and solves jigsaw task to regularize the model. **MASF** (Dou et al., 2019) and MetaReg (Balaji et al., 2018) are method that regularizes the network in the meta-learning paradigm. **MMLD** (Matsuura & Harada, 2020) proposes to match the latent distribution in different domains without accessing the domain information. **Epi-FCR** (Li et al., 2019) and **D-SAM** (D'Innocente & Caputo, 2018) is a method that fuse the shared and domain-specific layer to regularized the learned shared parameters can adapt to each domain-specific layer. Our method shares the same spirit but with a more effective regularization on domain invariant representation. **DSON** (Seo et al., 2019) is relative to our method in the normalization part, which separates the normalization layer with domain-specific. However, DSON ensembles the normalization layer during inference, which largely increases the inference time.

4.2 PACS DATASET

Table 1: Experimental results on PACS dataset (* denotes the model selected on the target domain).

Network	Method	A	C	P	S	Average
ResNet -18	ERM	77.78	73.93	**96.23**	73.43	80.34
	JigGen	79.42	75.25	96.03	71.35	80.51
	MASF	80.29	77.17	94.99	71.69	81.03
	IRM	80.60	73.64	95.63	76.64	81.63
	MetaReg*	**83.70**	**77.20**	95.50	70.30	81.70
	MMLD	81.28	77.16	95.87	72.29	81.83
	Epi-FCR	82.10	77.00	93.90	73.00	81.50
	InvarNorm	80.62	76.87	95.45	**79.33**	**83.07**
ResNet -50	ERM	85.79	79.40	96.83	82.13	86.03
	MASF	82.89	80.49	95.01	72.29	82.67
	MetaReg*	**87.20**	79.20	**97.60**	70.30	82.67
	InvarNorm	86.08	**80.64**	96.04	**82.92**	**86.42**

PACS is a standard domain generalization benchmark built by (Li et al., 2017). It consists of four different style domains: Photo, Art Painting , Cartoon, and Sketch with 9,991 images and 7 image recognition categories. We use the standard train-val split with 90 % for training and 10% for evaluation and select the best-performed model in the training data validation set.

Results In Table 1, we report the experimental results on the unseen target domain and compare it with state-of-the-art in the PASC dataset. We draw the following observations. First, comparing our approach InvarNorm with existing domain generalization methods in terms of Average Acc,

InvarNorm leads to the best results with 83.07% Acc, which illustrates the effectiveness of our approach in enhancing domain generalization ability. Second, by checking the results on the architecture ResNet-18 and ResNet-50, we can get that our approach consistently stabilize the batch normalization in both shallow and deep networks. Compared with IRM, our method outperforms it with 1.37% on average, which demonstrates the effectiveness in learning an invariant representation based on the normalization layer. Our InvarNorm also beats other meta learning (Dou et al., 2019) or augmentation based methods. Note that our InvarNorm is orthogonal to them and can be applied to both meta-learning and data augmentaiton framework.

4.3 OFFICE-HOME

Office-home is another large-scale generalization benchmark with around 15,500 images and 65 categories. Similar to PACS, the four domains in office-home: Artistic Images, Clip art, Product, and Real-World are collected with different style of images. Compared with PACS, Office-home is a difficult task, which has more training data and classes with large domain gap.

Table 2: Experimental results on the office-home dataset

Network	Method	A	C	P	R	Average
ResNet -18	ERM	58.46	41.51	70.71	73.35	61.01
	IRM	59.02	41.33	68.27	72.55	60.29
	D-SAM	58.03	44.37	69.22	71.45	60.77
	JiGen	53.04	**47.51**	**71.47**	72.79	61.20
	InvarNorm	**60.41**	43.98	70.64	**74.32**	**62.46**
ResNet -50	ERM	66.03	48.25	74.37	78.00	66.67
	InvarNorm	**67.73**	**49.14**	**75.32**	**79.01**	**67.51**

Results We report the domain generalization results on Office-home dataset compared with recent state-of-the-arts in Table 2. We can see that our InvarNorm consistently outperforms our baseline ERM on different network architectures, which proves the flexibility of our InvarNorm. Compared with D-SAM and JiGen (Carlucci et al., 2019) that implicitly regularize the network to increase its domain generalization ability, the improvement of InvarNorm shows that learning a domain invariant representation on the normalization space is effectiveness.

4.4 VLCS

VLCS is a classic benchmark for domain generalization task, with collecting data from four datasets : 1). PASCAL VOC 2007 (V), 2). LabelME (L), 3). Caltech, 4). SUN09. The training pipeline is following (Fang et al., 2013) that randomly divides training domains into 70% training and 30% for evaluation. Since our method focuses on addressing the problem in normalization layer, we replace the AlexNet in the conventional methods by ResNet-18 architecture.

Table 3: Experimental results on VLCS dataset

Method	V	L	C	S	Average
ERM	73.40	59.92	97.35	68.35	74.75
IRM	74.11	60.62	97.13	**68.57**	75.08
InvarNorm	**74.92**	**62.71**	**97.55**	68.31	**75.87**

Results In Table 3, we can draw similar observations with the results in PACS and Office-home that our InvarNorm effectively enhances model generalization ability compared with baseline method.

4.5 ABLATION STUDY

Marginal distribution shift. To measure how well of our InvarNorm in addressing the negative effect of marginal distribution shift to the normalization layer, we compare the performance gain of the source trained model after replacing the normalization statistics with the target domain. AlignNorm is another baseline that directly minimizes the normalization statistics between different domains, leading to the same normalization statistics in training domains. In Table 4, we can find that AdaBN

consistently improves the source trained model, which illustrates the change of normalization statistics leads to a performance drop. By checking the performance gain of network with and without AdaBN, we see that AdaBN brings ERM 1.81% improvement while AdaBN only improves InvarNorm with 0.72%, which illustrates that our InvarNorm is less sensitive to marginal distribution shift.

Table 4: Comparison the performance gain of AdaBN on different models.

Method	A	C	P	S	Average	Improv
ERM	77.78	73.93	96.23	73.43	80.34	-
ERM (AdaBN)	80.12	76.84	96.89	74.78	82.15	+1.81
AlignNorm	79.93	72.65	94.51	77.39	81.12	
AlignNorm (adabn)	80.56	75.42	95.21	78.23	82.35	+1.22
InvarNorm	80.62	76.87	95.45	79.33	83.07	-
InvarNorm (AdaBN)	81.03	77.92	96.23	79.98	83.79	+0.72

Comparison with different Normalization methods. To analyze the effectiveness of our InvarNorm in stabilizing batch normalization, we further conduct experiments to compare our InvarNorm with other normalization method, including (Chang et al., 2019), (Pan et al., 2018), (Luo et al., 2019), (Seo et al., 2019). The results of (Chang et al., 2019), (Pan et al., 2018), (Luo et al., 2019), (Seo et al., 2019) are directly copied from (Seo et al., 2019). Note that (Pan et al., 2018), (Luo et al., 2019), (Seo et al., 2019) contain instance normalization layer, which can alleviate the style shift to some extent. We follow the way in (Seo et al., 2019) to insert the instance normalization (Ulyanov et al., 2016) to our InvarNorm. Table 5 represents the results comparing with different normalization methods. We can see that our InvarNorm consistently outperforms the normalization layer without IN. By comparing the method with IN, InvarNorm also gets the best results and achieves similar results with the ensemble version of DSON in which the shared normalization layer can achieve similar results with ensemble of domain-specific normalization with our learning formulation.

Table 5: Comparison of different normalization on PACS and Office-home dataset. (sgl denotes the best results using the domain-specific branch, ens denotes the ensemble results in DSON, where ensemble operation requires 3 $\times$ inference time compared with a single model.)

Method	P	A	C	S	Avg	A	C	P	R	Avg
DSBN	78.6	66.2	95.5	70.2	77.6	59.0	45.0	**72.7**	72.0	62.2
IBN	75.3	73.0	92.0	77.4	79.4	55.4	44.8	68.3	72.0	60.1
SN	82.5	76.8	93.5	80.8	83.4	54.1	45.0	64.5	71.4	58.8
DSON(sgl)	78.7	75.7	95.4	79.5	82.3	-	-	-	-	-
DSON(ens)	**84.7**	**77.7**	95.9	**82.2**	**85.1**	59.4	**45.7**	71.8	**74.7**	**62.9**
BN	80.0	74.3	94.6	74.5	80.9	58.4	41.5	70.7	73.4	61.0
InvarNorm	81.7	75.5	96.1	79.0	83.1	60.9	44.0	70.6	74.3	62.5
BN+IN	81.5	76.4	**96.7**	80.3	83.7	56.3	41.3	67.5	73.7	59.7
InvarNorm + IN	83.5	77.5	96.2	81.7	84.8	**61.9**	43.9	70.5	74.5	62.8

5 CONCLUSION

We propose a new learning formulation and training pipeline in learning an invariant neural network based on batch normalization. Specifically, our formulation guides the neural network to learn invariant representation by penalizing the normalization statistics change due to the marginal distribution shift and rescaling parameter change due to the conditional distribution shift. InvarNorm provides a new perspective in identifying the the domain bias representation and robustify the representation to only contain invariant features. Our Extensive experimental results on three domain generalization benchmarks clearly illustrate that an invariant neural network can be learned by our proposed approach.

REFERENCES

Kartik Ahuja, Karthikeyan Shanmugam, Kush R. Varshney, and Amit Dhurandhar. Invariant risk minimization games. *ICML*, 2020.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *Arxiv*, abs/2003.12815, 2019.

Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. Metareg: Towards domain generalization using meta-regularization. In *NeurIPS*, 2018.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Vaughan. A theory of learning from different domains. *Machine Learning*, 79:151–175, 05 2010. doi: 10.1007/s10994-009-5152-4.

Fabio M. Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *CVPR*, 2019.

Woong-Gi Chang, Tackgeun You, Seonguk Seo, Suha Kwak, and Bohyung Han. Domain-specific batch normalization for unsupervised domain adaptation. In *CVPR*, 2019.

Antonio D'Innocente and Barbara Caputo. Domain generalization with domain-specific aggregation modules. In *GCPR*, 2018.

Qi Dou, Daniel C. Castro, Konstantinos Kamnitsas, and Ben Glocker. Domain generalization via model-agnostic learning of semantic features. In *NeurIPS*, 2019.

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

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35, 2016.

Rui Gong, Wen Li, Yuhua Chen, and Luc Van Gool. Dlow: Domain flow for adaptation and generalization. In *CVPR*, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR)*, 2016.

Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, 2017.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *PMLR*, 37:448–456, 07–09 Jul 2015.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019.

Hae Beom Lee, Taewook Nam, Eunho Yang, and Sung Ju Hwang. Meta dropout: Learning to perturb latent features for generalization. In *ICLR*, 2020.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Deeper, broader and artier domain generalization. In *ICCV*, 2017.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Learning to generalize: Meta-learning for domain generalization. In *AAAI*, 2018a.

Da Li, Jianshu Zhang, Yongxin Yang, Cong Liu, Yi-Zhe Song, and Timothy M. Hospedales. Episodic training for domain generalization. In *ICCV*, 2019.

Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex Kot. Domain generalization with adversarial feature learning. In *CVPR*, 2018b.

Ya Li, Mingming Gong, Xinmei Tian, Tongliang Liu, and Dacheng Tao. Domain generalization via conditional invariant representations. In *AAAI*, 2018c.

Ya Li, Xinmei Tian, Mingming Gong, Yajing Liu, Tongliang Liu, Kun Zhang, and Dacheng Tao. Deep domain generalization via conditional invariant adversarial networks. In *ECCV*, 2018d.

Yanghao Li, Naiyan Wang, Jianping Shi, Xiaodi Hou, and Jiaying Liu. Adaptive batch normalization for practical domain adaptation. *Pattern Recognition*, 80:109–117, 2018e.

Ping Luo, Jiamin Ren, Zhanglin Peng, Ruimao Zhang, and Jingyu Li. Differentiable learning-to-normalize via switchable normalization. *ICLR*, 2019.

Toshihiko Matsuura and Tatsuya Harada. Domain generalization using a mixture of multiple latent domains. In *AAAI*, 2020.

Hyeonseob Nam and Hyo-Eun Kim. Batch-instance normalization for adaptively style-invariant neural networks. In *NeurIPS*, 2018.

Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. Reducing domain gap via style-agnostic networks. *ArXiv*, abs/1910.11645, 2019.

Xingang Pan, Ping Luo, Jianping Shi, and xiaoou Tang. Two at once: Enhancing learning and generalization capacities via ibn-net. In *ECCV*, 2018.

Vihari Piratla, Praneeth Netrapalli, and Sunita Sarawagi. Efficient domain generalization via common-specific low-rank decomposition. *ArXiv*, abs/2003.12815, 2020.

Seonguk Seo, Yumin Suh, Dongwan Kim, Jongwoo Han, and Bohyung Han. Learning to optimize domain specific normalization for domain generalization. *ArXiv*, abs/1907.04275, 2019.

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

Hung-Yu Tseng, Hsin-Ying Lee, Jia-Bin Huang, and Ming-Hsuan Yang. Cross domain few-shot classification via learned feature-wise transformation. In *ICLR*, 2020.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *ArXiv*, abs/1607.08022, 2016.

Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, 2017.

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

Ximei Wang, Ying Jin, Mingsheng Long, Jianmin Wang, and Michael I Jordan. Transferable normalization: Towards improving transferability of deep neural networks. In *Advances in Neural Information Processing Systems 32*, 2019.

Cihang Xie, Mingxing Tan, Boqing Gong, Jiang Wang, Alan L. Yuille, and Quoc V. Le. Adversarial examples improve image recognition. In *CVPR*, 2020.